Studies in Computational Intelligence 1030

Nadia Nedjah
Ahmed A. Abd El-Latif
Brij B. Gupta
Luiza M. Mourelle *Editors*

Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities

Springer

Studies in Computational Intelligence

Volume 1030

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland

The series "Studies in Computational Intelligence" (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Indexed by SCOPUS, DBLP, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

More information about this series at https://link.springer.com/bookseries/7092

Nadia Nedjah · Ahmed A. Abd El-Latif ·
Brij B. Gupta · Luiza M. Mourelle
Editors

Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities

Springer

Editors
Nadia Nedjah
State University of Rio de Janeiro
Rio de Janeiro, Brazil

Brij B. Gupta
Department of Computer Science
and Information Engineering
Asia University
Taichung, Taiwan

Ahmed A. Abd El-Latif
Mathematics and Computer Science
Department, Faculty of Science
Menoufia University
Menofia, Egypt

Luiza M. Mourelle
State University of Rio de Janeiro
Rio de Janeiro, Brazil

ISSN 1860-949X ISSN 1860-9503 (electronic)
Studies in Computational Intelligence
ISBN 978-3-030-96736-9 ISBN 978-3-030-96737-6 (eBook)
https://doi.org/10.1007/978-3-030-96737-6

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

In the early days of AI, Artificial Intelligence (AI) and Robotics were inextricably linked, but have since separated. One of the first ambitions of artificial intelligence was to create embodied intelligent systems. However, such a goal has proven to be rather difficult, and researchers have separated its several unique elements and concentrated on progressing on each independently; this has resulted in AI and robotics evolving as disparate study lines with minimal cross-pollination of ideas. With breakthroughs in both sectors, there is increasing interest in integrating the two disciplines in order to generate a separate trend in integrated AI and robotics. AI in smart cities will play a significant part in advancing urbanization's goal of sustainable development by equipping cities with advanced features that enable residents to live, stroll, shop, and enjoy a more secure and comfortable lifestyle in such an environment.

With the greater responsibility placed on AI and robots in the workplace, the danger of a cyberattack increases proportionately. Cybercriminals, who may either locate an open window in the cloud and hack into the pepper robots or bring the financial markets to a halt, are perhaps the most worrying threat. The implications of a cyberattack might be catastrophic for organizations that face this criminal danger in the future. Not only are businesses exposed to direct financial losses, but they are also vulnerable to monetary fines for non-compliance with data privacy regulations. This book synthesizes the state of the art and practise in the emerging subject of integrated AI and robotics for cybersecurity and critical infrastructure in smart cities, highlighting the important directions in which machine intelligence is currently evolving.

This book is intended to provide a relevant reference for students, researchers, engineers, and professionals working in this particular area or those interested in grasping its diverse facets and exploring the latest advances on Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities.

We would like to sincerely thank the authors of the contributing chapters as well as reviewers for their valuable suggestions and feedback. The editors would like to thank Dr. Thomas Ditsinger (Springer, Editorial Director, Interdisciplinary Applied Sciences), Professor Janusz Kacprzyk (Series Editor in Chief), and Ms. Rini Christy

Xavier Rajasekaran (Springer Project Coordinator), for the editorial assistance and support to produce this important scientific work. Without this collective effort, this book would not have been possible to be completed. We hope you will enjoy this book and this amazing research field of Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities.

Rio de Janeiro, Brazil — Nadia Nedjah
Menofia, Egypt — Ahmed A. Abd El-Latif
Taichung, Taiwan — Brij B. Gupta
Rio de Janeiro, Brazil — Luiza M. Mourelle
December 2021

Contents

Deep Learning in Robotics for Strengthening Industry 4.0.: Opportunities, Challenges and Future Directions 1
Kriti Aggarwal, Sunil K. Singh, Muskaan Chopra, Sudhakar Kumar, and Francesco Colace

The Design of a Pheromone-Based Robotic Varroa Trap for Beekeeping Applications 21
Johannes Meister, Kim Ho Yeap, Magdalene Wan Ching Goh, Humaira Nisar, Johannes Fischer, and Hans Meier

Walk Through Event Stream Processing Architecture, Use Cases and Frameworks Survey 57
Passent el-Kafrawy and Mohamed Bennawy

Artificial Intelligence for Cybersecurity: Recent Advancements, Challenges and Opportunities 73
Veenu Rani, Munish Kumar, Ajay Mittal, and Krishan Kumar

Privacy and Security Concerns in Edge Computing-Based Smart Cities 89
Ashutosh Kumar, Aditya Upadhyay, Neha Mishra, Srawan Nath, Kalu Ram Yadav, and Gajanand Sharma

A Taxonomy of DDoS Defense Mechanism in Software Defined Networking (SDN) 111
Jasmeen Kaur Chahal, Vidhyotma Gandhi, and Payal Kaushal

Developing Instrument for Investigation of Blockchain Technology 123
Dmitry Kushnir, Maxim Kovtsur, Ammar Muthanna, Anastasiia Kistruga, Mark Akilov, and Anton Batalov

An IoT-Based System for Expert User Supporting to Monitor, Manage and Protect Cultural Heritage Buildings 143
Mario Casillo, Francesco Colace, Angelo Lorusso, Francesco Marongiu, and Domenico Santaniello

Commercializing Blockchain in a Smart City: Autonomous Vehicles, Cryptocurrency, and Contract Law 155
Nidhee Kamble

Finding All Shortest Meaningful Meta-Paths Between Two Vertices of a Secured Large Heterogeneous Information Network Using Distributed Algorithm 171
Phuc Do

Creation of a Dataset Modeling the Behavior of Malware Affecting the Confidentiality of Data Managed by IoT Devices 193
Alberto Huertas Celdrán, Pedro Miguel Sánchez Sánchez, Fabio Sisi, Gérôme Bovet, Gregorio Martínez Pérez, and Burkhard Stiller

Anomaly Detection on Static and Dynamic Graphs Using Graph Convolutional Neural Networks 227
Amani Abou Rida, Rabih Amhaz, and Pierre Parrend

Secure Medical Data Sharing Through Blockchain and Decentralized Models 249
Francesco Colace, Massimo De Santo, Francesco Marongiu, Domenico Santaniello, and Alfredo Troiano

Deep Learning in Robotics for Strengthening Industry 4.0.: Opportunities, Challenges and Future Directions

Kriti Aggarwal, Sunil K. Singh, Muskaan Chopra, Sudhakar Kumar, and Francesco Colace

Abstract The twenty-first century is undergoing a fundamental transition in the way things are created and services are provided. The digitalization of services has resulted in a compelling, major change in the sector, which is now referred to as Industry 4.0. One of the major changes embracing this industrial revolution is the development of robotics. Today, with the tasks becoming increasingly complex, robots are more trusted to perform the tasks with accuracy. They are rapidly substituting human skills in terms of speed, accuracy, and replaceability. Over the years, they have made vital contributions to some of the major industries, like 3D printing, autonomous vehicles, computer chip building, health and safety, agriculture, and so on. Robotics is a key Industry 4.0 technology that gives significant possibilities in the realm of production. However, with the introduction of Industry 4.0, it has become increasingly impossible for a corporation to remain relevant without incorporating some type of intelligent technology. Big data, which is generated by a wide range of sensors, necessitates complicated systems capable of distilling usable information and making intelligent judgments. This is where technology such as artificial intelligence and deep learning may help. The major result of combining these technologies with modern robotics is the creation of intelligent factories that are very powerful, safe, and cost-effective. The current research discussed the function of deep learning in robots for bolstering Industry 4.0. The authors discussed the various ways in which deep learning algorithms may be employed to improve the performance of robotics systems. Object identification, robotic grip, acoustic modelling, and motion control

K. Aggarwal (✉) · S. K. Singh · M. Chopra · S. Kumar
Chandigarh College of Engineering & Technology, Chandigarh, India

S. K. Singh
e-mail: sksingh@ccet.ac.in

S. Kumar
e-mail: sudhakar@ccet.ac.in

F. Colace
University of Salerno, Salerno, Italy
e-mail: fcolace@unisa.it

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_1

are the four key subjects covered in the article. A subsequent study discusses some of the significant problems for the same.

Keywords Deep learning · Robotics · Neural networks · Object grasping · Object detection · Motion control · Acoustic modelling

1 Introduction

The term "robot" was first introduced to the world in the year 1920, in Karel Capek's play called "Rosumovi Univerzáln Roboti". Traditional robots were designed to perform single, fixed, high-speed actions, and they required a highly experienced human staff to operate and maintain them. However, today with the integration of various technologies, robots have become capable of preforming multiple tasks efficiently and accurately. Robotic technology, which has made an indispensable contribution to modern industry, has seen significant development in recent years as a result of the advent and popularity of Industry 4.0 [1–3].

Industry 4.0 is a digital revolution that is taking place in our time, with the goal of digitising the whole production process with minimum human or physical interaction. The goal is to include as many sectors as possible while adapting and improving current technology to better meet the objectives of digital manufacturing. One of the most significant developments brought about by the Fourth Industrial Revolution is the advancement of robots. Figure 1 shows the challenges which industry 4.0 brings about in terms of technology and robotics.

Zero downtime and maximum efficiency are the goals of Industry 4.0-enabled robotics. From the development of robotics 1.0 to robotics 4.0, robots will grow

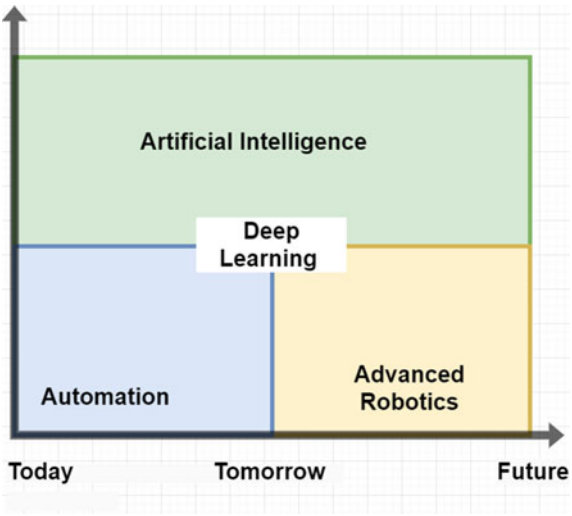


Fig. 1 Challenges in industry 4.0 (Smithers Piera)

less prone to disturbances as they employ more sensors and become more digitally linked. The next generation of robots and associated technologies are expected to play a larger role in meeting the dynamic demands of collaborative and intelligent production within the framework of technologies such as artificial intelligence (AI) and deep learning (DL) algorithms [4].

With the newer technologies and increased computer power, robots are ready to go beyond the lab and other special-purpose sectors such as manufacturing, computer chip building, health and safety, and agriculture into our daily life. Even though this is visible through rapidly growing research interest in the field of autonomous vehicles and self-driving cars, robotics as a whole can create even a larger impact [5].

A robot is a mechanism that can be modified by a computer and can conduct a set of complex activities autonomously. To manage the tremendous computational burden, the execution of all of these operations necessitates the use of parallel processing methods [6–8]. Robots can be constructed to look like humans, although most are intended to do tasks without consideration for aesthetics [9]. Humanoids are examples of robots that can be autonomous or semi-autonomous.

Robotics, or advanced mechanics, is the part of innovation that aligns with the planning, building, activity, and utilisation of robots as long as computational frameworks are utilised for their control, tangible input, and data handling [10]. These technologies are concerned with automated gadgets that can be used to substitute people in dangerous circumstances or in manufacturing operations. They are used to build systems that are capable of mimicking humans in terms of their behaviour, appearance, and cognition. Today, robotics has progressed from the laboratory to the real world. Figure 2 depicts the use of robots in several sectors.

Artificial intelligence and its subset, machine learning (ML), are used in many of the applications listed above. Robots might be invested with human-like capacities, including vision, contact, and temperature detection. Some are even equipped to settle on simple choices, and contemporary advanced mechanics research is pointed toward creating independent robots that would permit development and dynamics in an unstructured climate.

The role of deep learning in robotics is motivated primarily by the fact that it is broader than any other method of learning. Deep networks have been shown to be capable of high-level reasoning and abstraction. As a result, it is an excellent choice for robots in an uncontrolled environment. Furthermore, networks have become extremely successful as a result of developments in parallel computing and sophisticated, deep numerical libraries. For motion control, time-critical robotics jobs require a high-frequency response unit [11].

This paper reviews various works that have been done using deep learning to solve the visual object recognition problem and the robotic grasp problem in robotics. Further, aper is divided into two sections. The second section gives a brief overview of deep learning, while section three surveys the previous research work done in the field of deep learning.

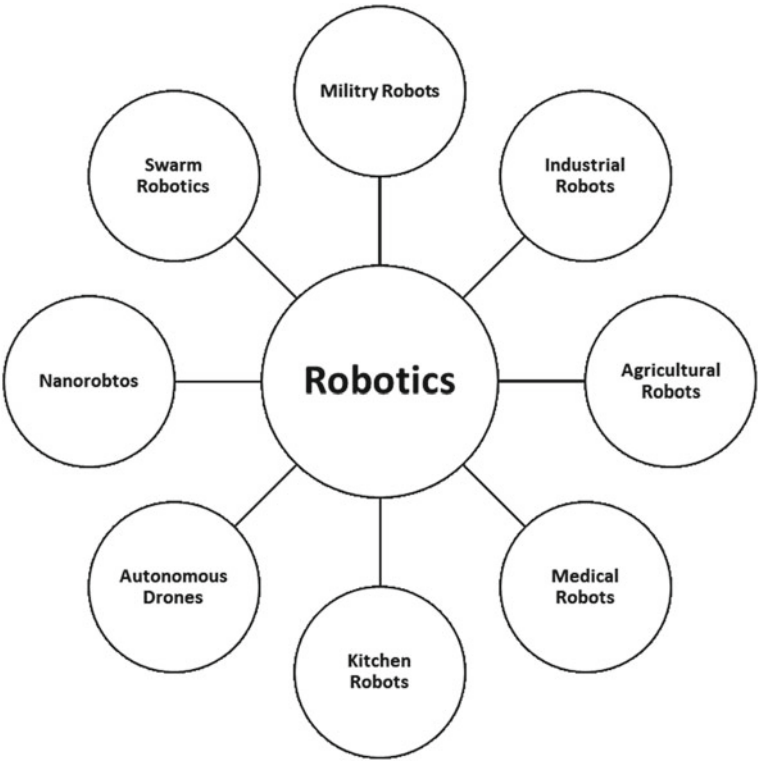


Fig. 2 Application of robotics in various domains

2 Introduction to Deep Learning

The study of training massive artificial neural networks (ANN) is known as "deep learning." As shown in Fig. 3, deep learning is a subset of ML and AI as shown in Fig. 3 [12]. It replicates how people acquire specific types of knowledge. Deep learning is also a critical component of data science, which also covers statistics and predictive modelling. Deep learning is particularly advantageous to data scientists who are entrusted with gathering, analysing, and interpreting massive volumes of data; deep learning speeds up and simplifies this process.

Since most deep learning methods rely on neural network topologies, they are commonly referred to as deep neural networks. Deep learning neural networks, also known as artificial neural networks, try to simulate the human brain using a mix of data inputs, weights, and bias. These aspects collaborate to effectively detect, categorise, and characterise things in data. Various deep learning models have been studies over the years to dolve day to day problems [13–15].

Deep neural networks are made up of numerous layers of interconnected nodes as shown in Fig. 4, each of which builds on the preceding layer to enhance and optimise

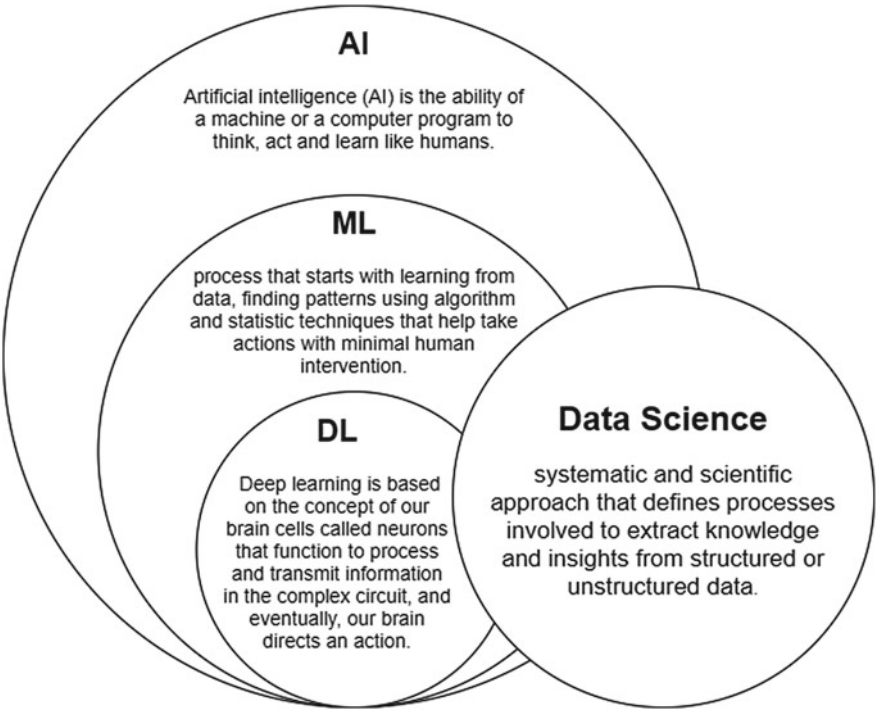

AI
Artificial intelligence (AI) is the ability of a machine or a computer program to think, act and learn like humans.
ML
process that starts with learning from data, finding patterns using algorithm and statistic techniques that help take actions with minimal human intervention.
DL
Deep learning is based on the concept of our brain cells called neurons that function to process and transmit information in the complex circuit, and eventually, our brain directs an action.
Data Science
systematic and scientific approach that defines processes involved to extract knowledge and insights from structured or unstructured data.

Fig. 3 Deep learning as a subset of Machine Learning, Artificial Intelligence and Data Science (Image *source*: [12])

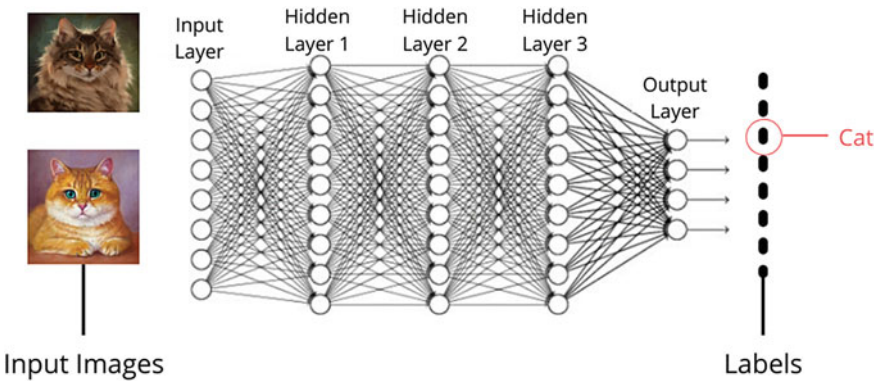


Fig. 4 Illustration of a deep learning neural networks [10]

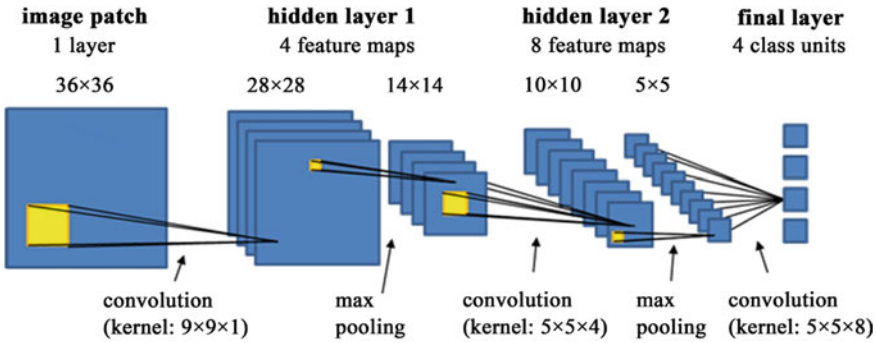


Fig. 5 Working of a convolutional neural network (Image *source* what is deep learning? IBM)

the prediction or categorization [16]. Because nodes collect and recombine input from previous levels, as you move through the neural network, the more complex the traits your nodes can recognise. This is referred to as feature hierarchy, which is a layer-by-layer hierarchy of increasing complexity [17].

The two most noticeable layers of a deep neural network are the information and yield layers. The profound learning model ingests information for handling in the info layer, and the last forecast or order is acted upon in the output layer.

The well-known deep learning methods are extremely sophisticated, and many types of neural networks exist to solve specific issues or datasets. Some of the popular deep learning techniques include ANNs, Recurrent Neural Networks (RNNS), and Convolutional Neural Networks (CNNs or ConvNets).

CNNs are widely used in image detection, computer vision, and image classification applications [18]. They can recognise properties and patterns within an image, allowing tasks such as object detection and recognition to be carried out. In 2015, for the first time, a CNN outperformed a human in an item recognition challenge [19]. Figure 5 depicts the primary algorithm [20].

Typically, the concealed layer has two separate layers. It is the first stage that results from the previous layer's local packing. Here, trainable weights are contained inside the core. The second stage is the maximum assembling stage. In this stage only the most responsive units of the first stage are maintained. After multiple concealments, the last layer usually has a completely linked structure. It obtains data from the preceding layer's units in each of those units, and each of those units has a unit for each class the network expects.

3 Deep Learning in Robotics

Robotics provides a variety of unique obstacles for learning algorithms. This can be seen in Fig. 6 which describes the seven most researched challenges in the field of robotics [21]. Robots must execute a broad variety of jobs. This serves as the

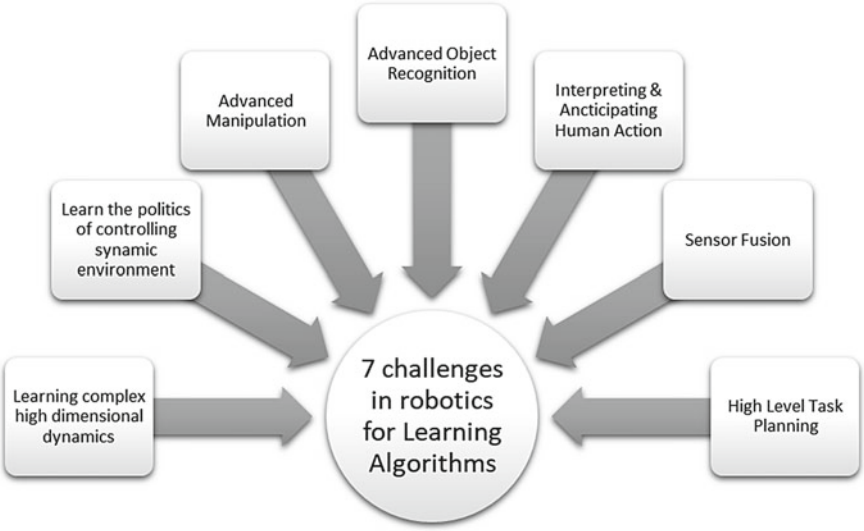

Advanced Object Recognition
Advanced Manipulation
Interpreting & Ancticipating Human Action
Learn the politics of controlling synamic environment
Sensor Fusion
7 challenges in robotics for Learning Algorithms
Learning complex high dimensional dynamics
High Level Task Planning

Fig. 6 Seven challenges in robotics for learning algorithms

first challenge for learning algorithms in robotics. It typically takes a lengthy or difficult time to code totally new learning algorithms and characteristics for each activity. Second, in the actual world, robots must deal with a considerable quantity of diversity, which many learning algorithms find challenging to deal with. Finally, because most robotic applications are time constrained, learning methods must be applied quickly in order to be effective.

As a result, roboticists are devising novel solutions, such as exploiting training data through digital manipulation, automating training, and deploying several DNNs to increase performance and minimise training time.

Modern deep learning approaches preserve the benefits of neural networks while overcoming their weakness using new algorithms and network topologies. Not only these, the computational performance of deep learning algorithms grows exponentially making them more suitable for large datasets [11] Deep learning has been used to a wide range of problems because of its efficacy as general non-linear learners [22], including natural language processing [23] visual recognition [24, 25], acoustic modelling [26], and many more. Recurrent deep networks have proved especially successful for time-dependent applications like speech recognition [27] and text production [28].

As shown in Fig. 7, the present paper discusses four main applications of deep learning in Robotics–Object detection, Acoustic Modelling, Object Grasping/Robotic Grasp/Motion Control.

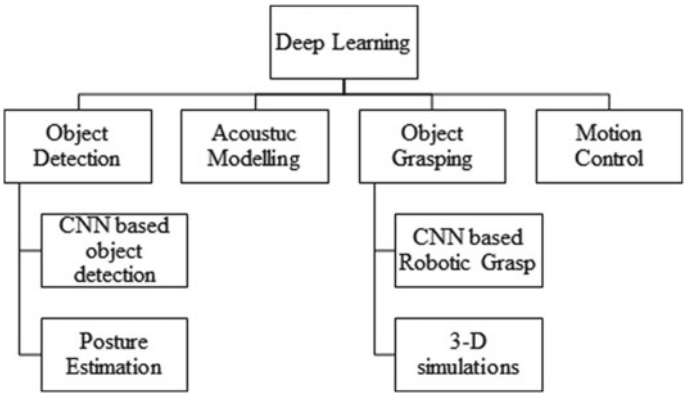


Fig. 7 Deep learning application for robotics

3.1 Object Detection

Autonomous object grabbing is seen as a fundamental ability for intelligent robots. A multi-robot sorting system, for example, can help individuals better manage household rubbish while also reducing environmental pollution. The foundation of robot grasping is object identification, and the convolution neural network has demonstrated great effectiveness in recognising objects in pictures. Figure 8 shows the object detection (people) [29]. Two tasks are involved in the object detection as follows:

1. Pre-processing data from real-world sensors to make it more useful by many subsystems [30].
2. Features such as corners, edges, and so forth can be extracted from data by using feature detection.

Object recognition and classification have always proven difficult owing to differences in views, variable picture sizes, and changeable lighting conditions. One such option is to use a neural network that has been trained to recognise and categorise

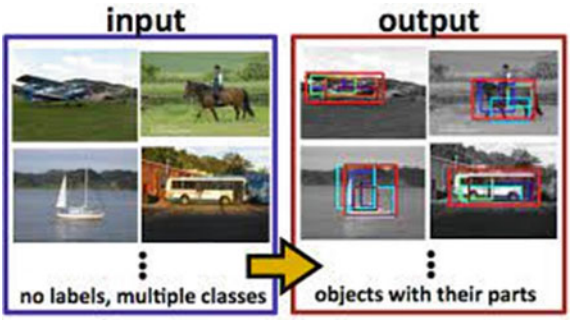


Fig. 8 Objects detection [11]

things. Object detection algorithms are further subdivided into one-level and two-level approaches. Several algorithms and research work done in this area has been studied in the Table 1.

The two stage object detection methods include the The RCNN series, which includes fast-RCNN, RCNN, faster-RCNN, Forest R-CNN and random faster R-CNN. While the RCNN, fast RCNN and faster RCNN are common models by Girshick that use two-level approaches [41]. Forest RCNN is a Large-Vocabulary and Long-Tailed Visual Recognition approach. It is capable of achieving 11.5% and 3.9% on the rare categories and overall categories, respectively.

When compared to two-stage approaches, one-stage methods eliminate the requirement for pre-processing for region suggestion, resulting in a quicker model that is largely appropriate for robot mechanics object identification. YOLO [42] is a forerunner of the one-stage technique. It utilises the entire image as the input to the network. The network then delivers the bounding box and appropriate category positions directly. SSD [43], based on VGG [44], provides a balance of speed and accuracy by utilising the concept of anchors.

Other YOLO based algorithms are YOLOv3, and LF-YOLO. While YOLO v3 has mAP of 93.88%, LF-YOLO has AP of 0.9197 with IoU threshold of 0.5.

3.2 Object Grasping or Robotic Grasp

As defined by Mahler et al. [45] grip detection refers to the capability of identifying gripping spots or grasping postures within an image. Figure 9 illustrates how a robotic right hand can lift an object with its fingers, and thus hold it firmly in place [46]. In addition to visually detecting objects within our surroundings, the human eye also helps us figure out how to approach and pick up objects. A robotic system can also utilize visual perception sensors to provide data about the climate that can be processed into useful information.

Traditionally, grasp detection needed specialised human expertise to analytically develop the algorithm for the task, but this is a time-consuming and difficult technique. Things gripping is a challenging issue due to a variety of things such as varied item forms and limitless object positions. To produce good outcomes, successful robotic grasping systems must be able to overcome this barrier [47]. Deep Learning methods for object detection are studied in the Table 2.

A sliding window detection framework is the most commonly used solution for 2-D robotic grab prediction. A classifier is employed in this framework to forecast whether a tiny portion of a picture being used as an input has a good conceivable grip on an item. The classifier is applied to a number of areas patch-wise on the picture, and regions with high scores are regarded as having good conceivable grip. Lenz et al. [48] employed this strategy as a classifier using convolutional neural networks and achieved a 75% accuracy. One significant disadvantage of their work is that it operates at a rate of 13.5 s per frame. In real-time applications, this is sluggish for a robot to determine where to move its EOAT.

Table 1 Object detection

S. No	Refs.	Dataset	Model name	Technique	Performance (mAP)
1.	[31]	PASCAL VOC 2010 200-class ILSVRC2013	RCNN	Two stage object detection	53.7% 31.4%
2.	[32]	PASCAL VOC 07	Fast R-CNN	Two stage object detection	66.9%
3.	[33]	Microsoft COCO object detection dataset	Faster R-CNN	Two stage object detection	41.5
4.	[34]	Large vocabulary dataset LVIs	Forest R-CNN	Large-vocabulary and long-tailed visual recognition	11.5% on rare and 3.9% on overall categories
5.	[35]	DOTA dataset	Rotated Faster R-CNN	On the basis of Faster R-CNN, we add a regression branch to predict the oriented bounding boxes for ground target	74.56
6.	[36]	COCO dataset	YOLOv3	Single stage method, backbone Darknet-53	AP 33.0
7.	[37]	PASCAL VOC COCO, ILSVRC datasets	Single Shot Detector (SSD)	Single stage method based on VGG	74.3% 73.2% 63.4%
8.	[38]	data set provided by the 2017 Underwater Robot Target Catching Competition	Class-weighted YOLO network	YOLO V3 network	76.8%
9.	[39]	New test dataset 1. SBU Kinect Interaction Dataset (SBU) 2. BJKinect − Interaction Dataset (BJK)	1. Convolutional Neural Networks for human action recognition 2. YOLO v3 for hand-held object identification	1. ST-GCN-LSTM model is proposed by fusing the spatial temporal graph convolutional networks and long short term memory networks for human action recognition 2. YOLO v3 for hand-held object identification	1. 86.89%, 87.30%, 89.34%, 86.07% and 88.52% in 5 tests 2. mAP = 93.88%

(continued)

Table 1 (continued)

S. No	Refs.	Dataset	Model name	Technique	Performance (mAP)
10.	[40]	333-dataset-new test dataset called 333 images including 166 images downloaded on the internet and 167 images captured outdoors	The LF-YOLO-based real-time object detection	Octree-aided motion control scheme	0.9197 AP, IoU thresh = 0.5

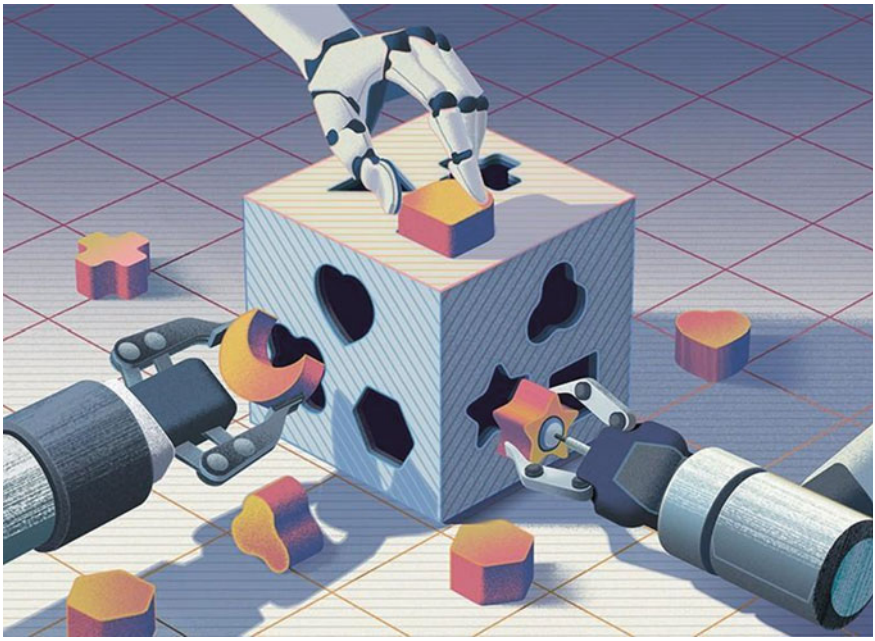


Fig. 9 Right hand robot mechanics holding a solid [46]

Another researcher used the AlexNet architecture to develop direct regression, regression along with classification and multigrasp detection algorithms [49]. The ResNet-50 architecture was used by Kumar S. to develop uni-modal SVM & RGB, uni-modal RGB, and uni-modal RG-D [50].

Some of the researchers working on RGB-D object detection include [51]'s generative grasping convolutional neural network (GG-CNN), [54]'s deep neural networks based on two-stage closed-loop, with penalty, and [53] VGG model based STEM-CaRFs (STructural EMbedding).

Table 2 Object grasping or robotic grasp

S. No	Refs.	Training dataset	Method	Architecture	Performance	
					Image wise split	Object wise split
1.	[48]	Cornell dataset	regression as classification approach	CNN	96.0%	96.1%
2.	[49]	Cornell dataset	Direct regression Regression + Classification MultiGrasp Detection	AlexNet	84.4% 85.5% 88.0%	84.9% 84.9% 87.1%
3.	[50]	Cornell Grasp Dataset	Uni-modal, SVM, RGB Uni-modal RGB Uni-modal RG-D	ResNet-50 Image wise accuracy	84.76% 88.84% 88.53%	84.47% 87.72% 88.40%
4.	[51]	The Cornell Grasping Dataset contains 885 RGB-D images of real objects, with 5110 human-labelled positive and 2909 negative grasps	Generative grasping convolutional neural network (GG-CNN)	Convolutional Neural Networks, RGB-D object recognition	–	83% on unseen objects with adversarial geometry and 88% on a set of household objects
5.	[52]	Cornell Grasp Detection Dataset and the unlabeled Washington RGB-D Object Dataset	Two-stage closed-loop, with penalty	Deep convolutional neural network (DCNN) model, RGB-D object recognition	82%	69% accuracy on 20 ground-truth novel objects
6.	[53]	Washington RGB-D object dataset and Cornell Grasp object dataset	STEM-CaRFs (STructural EMbedding)	VGG model, CNN-based, RGB-D object recognition	86.8%	86.2%

A large amount of effort has been made to locate good grasps using 3-D simulations [55–57]. These methods are effective, but the majority of them rely on a known 3-D model of the target item to compute an acceptable grip. General-purpose robots, on the other hand, should be able to grip unknown items in the absence of the object's 3-D representation. According to [58], deep learning has the potential for 3-D object

Table 3 Acoustic modelling

S. No	Refs.	Dataset	Model name	Technique	Performance
1.	[60]	(16,800 + 2400)*9 = 172,800 train sample images	VGG-Face CNN and multi-scale vectors from Curvelet decomposition	Net processing, multi-layer output vectors	99.5% accuracy for yaw angle estimation 99.7% for pitch angle estimation
2.	[61]	Total of 2,430,000 files selected	DCNN and end-to-end training to perform SSL	DL models for SSL tasks	Block accuracy of the median angle has a tolerance of ± 2.5
3.	[62]	Single cluster, 4-16 GB Tesla V100s	BERT-Large fine-tuning	SQuAD workload	1.2 million parameter feedforward neural network

recognition and posture estimation, but their studies only employed five items and their technique is computationally costly.

3.3 Acoustic Modeling

Acoustic is a word that is used in a variety of industries. Acoustics is challenged in many ways since acoustic physics refers to knowledge of the subject of mechanical waves that exist in many objects, such as gases, liquids, and solids. Acoustic categorization is divided into stages. Based on the data, the audio data can be categorised using unsupervised or supervised learning [59]. This may be accomplished by applying various deep learning models, such as neural networks, and feature extraction algorithms to extract features from audio data. Following that, the model's performance for acoustic scene categorization is tested. Acoustic models are used for more than just sound classification; they are also used for picture classification and other types of classification.

Several algorithms and research work done in this area have been studied in Table 3.

3.4 Motion Control

Deep learning motion control algorithms have recently been applied in a variety of scientific research. Classical control has accurately and analytically solved a wide range of robotic control challenges, allowing robots to perform complex manoeuvres. Traditional control theory, on the other hand, only handles the problem for a single scenario and an approximated robot model, making adaptation to changes in the

Table 4 Motion control

S. No	Refs.	Dataset	Model name	Technique	Performance
1.	[63]	2,600 static images from 0° and 90° camera view angles	DNN based 3D coordinate data captured and classification model	Keras	Accuracy 0° → (91.64%) and 90° → (90.86%)
2.	[64]	3D gesture recognition or sign language datasets captured by Kinect	DBN, CNN, HOG + SVM	Kinect using deep belief nets (DBNs) CNN-LeNet-5 for hand posture recognition	DBN is 98.12%, while the CNN is 94.17%, and HOG + SVM is 87.58%
3.	[65]	8206 training samples from kinect sensor	RNN	–	40% reduction in the prediction error
4.	[66]	Object dataset is created in the simulator	CNN	Long short term memory (LSTM)	93% accuracy
5	[54]	Training dataset-ImageNet	DCNN	AlexNet	Accuracy of 96.6%

robot model and/or hostile settings problematic. Learning from experience is crucial in this situation since it may help you overcome many of the limits outlined.

Motion control consists of posture control and motion recognition. Many work has been done in both of these areas and are summarized in Table 4.

4 Challenges and Future Opportunities

In this decade, deep learning, the backbone of artificial intelligence, has become one of the most fascinating technologies. It has already made significant advances in the fields of automation, health-care and covid-19 detection [14, 67, 68, text analysis [69], fake news detection [70], data security [71], and robotics.

Some of the major challenges to deep learning adoption in robotics include the need for vast amounts of training data and lengthy training timeframes. Creating training data for physical systems may be time consuming and costly. Crowdsourcing training data using cloud robotics is one potential trend [72]. Yang's use of general-purpose cookery films for object and grip identification demonstrates that this data does not even have to come from other robots [73]. Local parallel processing [74] and advancements in raw processing speed have resulted in considerable improvements in training time.

Distributed computing allows for the allocation of more computer resources to a given issue [75], but it is constrained by communication rates [76]. There may also be computational approaches to improve the efficiency of the training process that

have yet to be identified. DNNs appear to be potent tools for robotics practitioners at the moment, but only time will tell whether they can truly supply the skills required for dexterous adaptation in broad contexts.

5 Conclusion

The present paper talked about the role of deep learning in the field of robotics. The authors talked about the possible areas where algorithms can be improved in robotics. The paper also surveyed recent deep learning approaches for solving problems in the field of object detection, robotic grasp, acoustic modelling, and motion control. With the coming of Industry 4.0, deep learning-based technologies and algorithms are being used to combine different skills like perception, visualization, etc., into a single robotic system. Deep Learning and neural networks can function on raw data collected from real-time sensors without the need for human interaction, possibly decreasing pre-engineering time by half. They are also capable of integrating multi-media and high-dimensional data. The progress has been explained over time, making it simple to get used to the dynamics without structures in which robots work. However, there are still various challenges in this field, some of which have been discussed in paper. Future of robotics lie in applying deep learning algorithms successfully along with artificial intelligence-based learning methods in order to deliver almost human-like robots.

References

1. Ullah, A.S.: Modeling and simulation of complex manufacturing phenomena using sensor signals from the perspective of Industry 4.0. Adv. Eng. Inf. **39,** 1–13 (2019)
2. Zezulka, F., Marcon, P., Bradac, Z., Arm, J., Benesl, T., Vesely, I.: Communication systems for industry 4.0 and the IIoT, IFAC-PapersOnLine **51**, 150–155 (2018)
3. Dalenogare, L.S., Benitez, G.B., Ayala, N.F., Frank, A.G.: The expected contribution of Industry 4.0 technologies for industrial performance. Int. J. Prod. Econ. **204**, 383–394 (2018)
4. Gupta, B.B., Tewari, A., Cvitić, I., et al.: Artificial intelligence empowered emails classifier for internet of things based systems in industry 4.0. Wireless Netw. 1–11 (2021)
5. Sharma, S., Singh, S., Panja, S.: Human factors of vehicle automation. In: Autonomous Driving and Advanced Driver-Assistance Systems (ADAS), Taylor & Francis Group (CRC Press), Chapter 15 (2021)
6. Singh, S.K., Madaan, A., Aggarwal, A., Dewan, A.: Design and implementation of a high performance computing system using distributed compilation. In: 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1352–1357 (2013). https://doi.org/10.1109/ICACCI.2013.6637374
7. Singh, S.K., Singh, R.K., Bhatia, M.: Design flow of reconfigurable embedded system architecture using LUTs/PLAs. In: 2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, pp. 385–390 (2012). https://doi.org/10.1109/PDGC.2012.6449851
8. Kumar, S., Singh, S.K., Aggarwal, N., Aggarwal, K.: Evaluation of automatic parallelization algorithms to minimize speculative parallelism overheads: an experiment. J. Discrete Math. Sci. Cryptog. **24**(5), 1517–1528 (2021). Taylor & Francis

9. Definition of "Robot". Oxford English Dictionary (2016)
10. Built In What is robotics? Types of Robots. https://builtin.com/robotics (2019)
11. Madan, R., Singh, S.K., Jain, N.: Signal filtering using discrete wavelet transform. Int. J. Recent Trends Eng. **2**(3), 96 (2009)
12. Deep Learning versus machine learning–what's the difference? Levity. (n.d.). https://levity.ai/blog/difference-machine-learning-deep-learning (2021). Retrieved 25 Oct 2021
13. Hammad, M., Iliyasu, A.M., Subasi, A., Ho, E.S., El-Latif, A.A.: A multitier deep learning model for arrhythmia detection. IEEE Trans. Instrum. Measurem. **70**, 1–9 (2021). https://doi.org/10.1109/tim.2020.3033072
14. Sedik, A., Hammad, M., Abd El-Samie, F.E., et al.: Efficient deep learning approach for augmented detection of Coronavirus disease. Neural Comput. Appl. 1–18 (2021)
15. Sedik, A., Iliyasu, A.M., Abd El-Rahiem, B., Abdel Samea, M.E., Abdel-Raheem, A., Hammad, M., Peng, J., Abd El-Samie, F.E., Abd El-Latif, A.A.: Deploying machine and deep learning models for efficient data-augmented detection of COVID-19 infections. Viruses **12**(7), 769 (2020). https://doi.org/10.3390/v12070769
16. Parmar, R.: Training deep neural networks. Medium. https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964 (2018). Retrieved 25 Oct 2021
17. P versus NP Solution–Advances in Computational Complexity, Status and Future Scope. https://doi.org/10.5120/ijca2019919465
18. Singh, I., Sunil, S.K., Kumar, S., Aggarwal, K.: Dropout-VGG based convolutional neural network for traffic sign categorization. In: The proceeding of 2nd Congress on Intelligent Systems (CIS 2021), Lecture Notes on Data Engineering And Communication Technologies. Springer Book (2021)
19. IBM Cloud Education. (n.d.). What is deep learning? IBM. https://www.ibm.com/cloud/learn/deep-learning. Retrieved 25 Oct 2021
20. Ghorbani, M.A., Zadeh, H.A., Isazadeh, M., et al.: A comparative study of artificial neural network (MLP, RBF) and support vector machine models for river flow prediction. Environ. Earth Sci. **75**, 476 (2016). https://doi.org/10.1007/s12665-015-5096-x
21. Punjani, A., Abbeel, P.: Deep learning helicopter dynamics models. In: IEEE International Conference on Robotics and Automation (ICRA), Seattle, pp. 26–30, 3–45 (2015). https://doi.org/10.1109/ICRA.2015.7139643
22. Bengio, Y.: Learning deep architectures for AI. FTML **2**(1), 1–127 (2009)
23. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: ICML (2008)
24. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
25. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML (2009)
26. Mohamed, A.-R., Dahl, G., Hinton, G.E.: Acoustic modeling using deep belief networks. IEEE Trans. Audio Speech Lang. Proc. **20**(1), 14–22 (2012)
27. Graves, A., Mohamed, A., Hinton, G.E.: Speech recognition with deep recurrent neural networks. In: ICASSP (2013)
28. Sutskever, I., Martens, J., Hinton, G.: Generating text with recurrent neural networks. In: ICML (2011)
29. Media, O.S.: (n.d.) Robotics vision processing: Object detection and tracking. Embedded Comput. Design. https://www.embeddedcomputing.com/application/industrial/robotics-vision-processing-object-detection-and-tracking (2021). Retrieved 25 Oct 2021
30. Singh, S.K., Kaur, K., Aggarawal, A.: Emerging trends and limitations in technology and system of ubiquitous computing. Int. J. Adv. Res. Comput. Sci. (IJARCS) **5**(7), 174–178 (2014)
31. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
32. Girshick, R.: Fast R-CNN. In: IEEE International Conference on Computer Vision (ICCV) (2015)

33. Ren, S., et al.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Patt. Anal. Mach. Intell. **39.6**, 1137–1149 (2017)
34. Wu, J., Song, L., Wang, T., Zhang, Q., Yuan, J.: Forest R-CNN: large-vocabulary long-tailed object detection and instance segmentation. In: Proceedings of the 28th ACM International Conference on Multimedia. Association for Computing Machinery, New York, NY, USA, pp. 1570–1578 (2020). DOI:https://doi.org/10.1145/3394171.3413970
35. Yang, S., Pei, Z., Zhou, F., Wang, G.: Rotated faster R-CNN for oriented object detection in aerial images. In: Proceedings of the 2020 3rd International Conference on Robot Systems and Applications, ICRSA 2020. Association for Computing Machinery, New York, NY, USA, pp. 35–39. https://doi.org/10.1145/3402597.3402605
36. Redmon, J., Farhadi, A.: YOLOv3: an Incremental Improvement. In: arXiv:1804.02767 2018
37. Liu, W., Anguelov, D., Erhan, D., et al.: SSD: single shot multi box detector. Journal (2016)
38. Zhang, L., Li, C., Sun, H.: Object detection/tracking toward underwater photographs by remotely operated vehicles (rovs). Future Gener. Comput. Syst. **126**, 163–168 (2022). https://doi.org/10.1016/j.future.2021.07.011
39. Liu, C., Li, X., Li, Q., Xue, Y., Liu, H., Gao, Y.: Robot recognizing humans intention and interacting with humans based on a multi-task model combining St-GCN-LSTM model and Yolo Model. Neurocomputing **430**, 174–184 (2021). https://doi.org/10.1016/j.neucom.2020.10.016
40. Shi, Q.-X., Li, C.-S., Guo, B.-Q., Wang, Y.-G., Tian, H.-Y., Wen, H., Meng, F.-S., Duan, X.-G.: Manipulator-based autonomous inspections at road checkpoints: Application of faster Yolo for detecting large objects. Defence Technol. (2021). https://doi.org/10.1016/j.dt.2021.04.004
41. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. https://arxiv.org/abs/1506.01497 (2016). Retrieved 9 Nov 2021
42. Redmon, J., et al.: You only look once: unified, real-time object detection. Comput. Vision Patt. Recog. IEEE (2016)
43. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: single shot multibox detector. In: ECCV (2016)
44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Comput. Sci. (2014)
45. Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Aparicio, J., Goldberg, K.: Dex-Net 2.0: deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In: Proceedings of the Robotics: Science and Systems, Cambridge, MA, USA, 12–16 (2017)
46. Hodson, R.: How robots are grasping the art of gripping. Nature News. https://www.nature.com/articles/d41586-018-05093-1 (2018). Retrieved 25 October 2021
47. Song, D., Huebner, K., Kyrki, V., Kragic, D.: Learning taskconstraints for robot grasping using graphical models. In: International Conference on Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ, pp. 1579–1585 (2010)
48. Lenz, I., Lee, H., Saxena, A.: Deep learning for detecting roboticgrasps. Int. J. Robot. Res. **34**(4–5), 705–724 (2015)
49. Redmon, J., Angelova, A.: Real-time grasp detection using convolutional neural networks. In: Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30, pp. 1316–1322 (2015)
50. Kumra, S., Kanan, C.: Robotic grasp detection using deep convolutional neural networks. In: Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28, pp. 769–776 (2017)
51. Morrison, D., Corke, P., Leitner, J.: Closing the loop for robotic grasping: a real-time, generative grasp synthesis approach. Journal (2018)
52. Wang, Z., Li, Z., Wang, B., Liu, H.: Robot grasp detection using multimodal deep convolutional neural networks. Adv. Mech. Eng. **8**(9) (2016)
53. Asif, U., Bennamoun, M., Sohel, F.A.: Rgb-d object recognition and grasp detection using hierarchical cascaded forests. IEEE Trans. Robot. (2017)

54. Wang, P., Liu, H., Wang, L., Gao, R.X.: Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. CIRP Ann. **67**(1), 17–20 (2018). https://doi.org/10.1016/j.cirp.2018.04.066
55. Bohg, J., Kragic, D.: Learning grasping points with shape context. Rob. Auton. Syst. **58**(4), 362–377 (2010)
56. Krainin, M., Curless, B., Fox, D.: Autonomous generation of com-plete 3d object models using next best view manipulation planning. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 5031–5037 (2011)
57. Le, Q.V., Kamm, D., Kara, A.F., Ng, A.Y.: Learning to graspobjects with multiple contact points. In: IEEE International Conference on Robotics and Automation(ICRA), pp. 5062–5069 (2010)
58. Yu, J., Weng, K., Liang, G., Xie, G.: A vision-based robotic grasping system using deep learning for 3d object recognition and pose estimation. In: IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1175–1180 (2013)
59. Cho, M., et al.: Unsupervised object discovery and localization in the wild: part-based matching with bottom-up region proposals. https://arxiv.org/pdf/1501.06170v3.pdf. (2015). Retrieved 25 October 2021
60. Han, Z., Song, W., Yang, X., Ou, Z.: Face pose estimation with ensemble multi-scale model and deep learning. In: Proceedings of the 2018 2nd International Conference on Deep Learning Technologies-ICDLT '18 (2018). https://doi.org/10.1145/3234804.3234822
61. Yalta, N., Nakadai, K., Ogata, T.: Sound source localization using deep learning models. J. Rob. Mech. **29**(1), 37–48 (2017). https://doi.org/10.20965/jrm.2017.p0037
62. Nagrecha, K.: Model-parallel model selection for deep learning systems. In: Proceedings of the 2021 International Conference on Management of Data (2021). https://doi.org/10.1145/3448016.3450571
63. Hung, J.-S., Liu, P.-L., Chang, C.-C.: A deep learning-based approach for human posture classification. In: Proceedings of the 2020 2nd International Conference on Management Science and Industrial Engineering (2020). https://doi.org/10.1145/3396743.3396763
64. Tang, A., Lu, K., Wang, Y., Huang, J., Li, H.: A real-time hand posture recognition system using deep neural networks. ACM Trans. Intell. Syst. Technol. **6**(2), 1–23 (2015). https://doi.org/10.1145/2735952
65. Zhang, J., Liu, H., Chang, Q., Wang, L., Gao, R.X.: Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly. CIRP Ann. **69**(1), 9–12 (2020). https://doi.org/10.1016/j.cirp.2020.04.077
66. Seker, M.Y., Tekden, A.E., Ugur, E.: Deep effect trajectory prediction in robot manipulation. Robot. Auton. Syst. **119**, 173–184 (2019). https://doi.org/10.1016/j.robot.2019.07.003
67. Elmisery, A.M., Sertovic, M., et al.: Cognitive privacy middleware for deep learning mashup in environmental IoT. IEEE Access **6**, 8029–8041 (2017)
68. Rahman, M.A., Hossain, M.S., Alrajeh, N.A., et al.: A multimodal, multimedia point-of-care deep learning framework for COVID-19 diagnosis. ACM Trans. Multimidia Comput. Comm. Appl. **17**(1s), 1–24 (2021)
69. Yen, S., Moh, M., Moh, T.S.: Detecting compromised social network accounts using deep learning for behavior and text analyses. Int. J. Cloud Appl. Comput. (IJCAC) **11**(2), 97–109 (2021)
70. Sahoo, S.R., et al.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Appl. Soft Comput. **100**, 106983 (2021)
71. Jiang, F., Fu, Y., et al.: Deep learning based multi-channel intelligent attack detection for data security. IEEE Trans. Sustain. Comput. **5**(2), 204–212 (2018)
72. Pratt, G.A.: Is a cambrian explosion coming for robotics? J. Econ. Perspect. **29**(3), 51–60 (2015)
73. Yang, Y., Li, Y., Fermüller, C., et al.: Robot learning manipulation action plans by watching unconstrained videos from the world wide web. In: 29th AAAI Conference on Artificial Intelligence (AAAI-15); Austin, TX (2015)
74. Oh, K., Jung, K.: GPU implementation of neural networks. Patt. Recog. **37**(6), 1311–1314 (2004)

75. Schmitz, A., Bansho, Y., Noda, K., et al.: Tactile object recognition using deep learning and dropout. In: 14th IEEE-RAS International Conferences on Humanoid Robots, pp. 1044–1050 (2014)
76. Jordan, M.I., Mitchell, T.M.: Machine learning: trends, perspectives, and prospects. Science **349**(6245), 255–260 (2015)

The Design of a Pheromone-Based Robotic Varroa Trap for Beekeeping Applications

Johannes Meister, Kim Ho Yeap, Magdalene Wan Ching Goh, Humaira Nisar, Johannes Fischer, and Hans Meier

Abstract We present the design of a Varroa destructor trap for beekeeping by using pheromones. The trap is to kill the main enemy of the European or North American honey bee, the so called Varroa mite or Varroa destructor, with vaporized pheromones. The literature review gives a summary about the actual methods for Varroa treatment. Furthermore, it includes methods which can be used to build up a Varroa trap system with a vaporizer and high voltage grid and some additional parts. First of all, an enclosure where the vaporizer and the high voltage grid can be installed, has to be designed and the high voltage grid installed. To use the high voltage grid, a circuit which is the source for the high voltage has to be designed. After this is done, a vaporizer has to be designed and a suitable temperature control for vaporizing the pheromones is developed. This is done with a negative temperature coefficient (NTC) resistor, a measurement circuit for the NTC resistor, the analog-to-digital converter (ADC) of a microcontroller and a switching circuit for the heating element. The last step includes the design of a PCB and the implementation of all software routines to get a working system. All necessary hardware parts are combined on a single PCB

J. Meister · J. Fischer · H. Meier
Fakultät Angewandte Natur- und Kulturwissenschaften, Ostbayerische Technische Hochschule Regensburg, Regensburg, Germany
e-mail: johannes.meister@st.oth-regensburg.de

J. Fischer
e-mail: johannes.fischer@oth-regensburg.de

H. Meier
e-mail: hans.meier@oth-regensburg.de

K. H. Yeap (✉) · H. Nisar
Faculty of Engineering and Green Technology, Universiti Tunku Abdul Rahman, Kampar, Perak, Malaysia
e-mail: yeapkh@utar.edu.my

H. Nisar
e-mail: humaira@utar.edu.my

M. W. C. Goh
School of Science and Technology, Wawasan Open University, Penang, Malaysia
e-mail: magdalenegoh@wou.edu.my

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_2

and all software parts are implemented and programmed into the microcontroller. Furthermore, this system is tested to ensure the vaporizing and the high voltage grid is working properly. The tests have shown that the vaporizer and therefore the temperature control performs as required. In addition, the high voltage generation and high voltage grid had been verified which means the desired functionality of these parts is also given. All in all, the system is a well working prototype which can be used for beekeeping.

Keywords Pheromone · Varroa · High voltage · Microcontroller · Negative temperature coefficient (NTC) resistor · Analog-to-digital converter (ADC) · Honey bee

1 Introduction

The so called Varroa mite (also called Varroa destructor or Varroa jacobsoni) is considered to be the main enemy or misery of honey bees in the whole world [8]. Originally, the Varroa was confined to the honey bees on the Asian continent (Apis cerana) [15]. But due to international trading, the Varroa has spread around the entire world and has also found its way to the European honey bee [14].

Because the Apis cerana is the original host of the mite, it built up several defense methods to reduce the impact of Varroa on their colonies. These methods are not adopted by the European honey bee so it is easier for the mites to exploit or influence their colonies [19].

Because honey bees play a major role in our ecological system and are also an economical part of many farmers (pollination of plants or crops) the Varroa mite endangers a lot of bee colonies which might lead to an imbalance of insects or more serious to severe problems in the world. Even the great physicist, Albert Einstein is said to have said following sentence: “First the bees die then the human being dies”.

This problem has brought up some solutions to impact the Varroa population in a bee keeping colony which are more or less effective but furthermore also affect the bees in a bad way.

Almost every person in the area of apiculture or bee keeping knows about the risk and dangers because of the Varroa destructor. Due to the fact that this enemy influences the honey income this is a massive problem for bee keepers. The main difficulty is to reduce the population of the Varroa without any or almost no influence on the bees in the colony. For successful reproduction the mites need to find a suitable host larva in the honey bees brood cells [23].

There are several approaches which can be used to overcome this mite problem. These concepts cover constructional changes in a beehive [17], with the help of heat in the beehive [5, 10] or with chemicals as described in Abou-Shaara and Radetzki [1, 14]. Despite being effective in containing the Varroa mites, these existing approaches exhibit certain disadvantages. The disadvantages of the approaches are elaborated

in detail in the subsequent section. There is therefore a need to develop a better approach–one which is able to overcome the drawbacks of the existing approaches.

In this chapter, the design of a robotic Varroa destructor trap for beekeeping is presented. A type of chemical fluid is applied in the proposed robotic cage. In contrast to the methods in Abou-Shaara and Radetzki [1, 14] the fluid is not used to destroy the mites. Rather, it will be used to create an attracting vapor from sexual pheromones [23]. The idea is to attract the Varroa mites to the vaporized pheromone and then, kill them with the help of a high voltage fence around the vapor source. Since robotic systems and artificial intelligence have been widely integrated into conventional systems to increase their efficiencies [4, 7, 9, 11, 16], the employment of a robotic system in this proposed approach shall be able to ameliorate the efficacy of entrapping the Varroa mites.

For this approach, it is necessary to build a housing for the high voltage fence, a high voltage source for this fence, a control system for vaporizing the fluid attractant at the right temperature, and a control unit with user interface for operating the system.

2 Existing Solutions

To find out why a new method for controlling the Varroa population is useful or valuable one must first compare the existing methods. Peter Rosenkranz describes in his paper that if the Varroa mite is detected in a country it is usual that bee scientist, veterinary authorities and extension services put a lot of efforts to control the spread of this parasite and also to prevent a crunch of the honey bee colonies [15].

It depends on the bee keepers on whether to take actions against the Varroa or not. The one who will not focus on controlling the Varroa could lose their bee colonies. Even if the situation stabilizes due to the control there are, mostly during winter, high losses of about 30% or more of honey bee population in apicultural hives [15]. In several papers or publications, a lot of Varroa treatment are described and therefore they have to be analyzed. Figure 1 shows the so-called integrated pest management (IPM) pyramid. As one can see, it is structured into three parts. At the bottom is the cultural treatment which focuses on reducing pest reproduction. The second part is the mechanical approach. This means the Varroa will be killed by physical means with traps or other methods. At the top of the pyramid is the use of chemical substances. It is subdivided into the group of soft chemical treatment with natural components, and the hard chemical treatment which requires the usage of synthetic compounds.

This figure also shows how the different treatments work. Started at the bottom by preventing the parasite it goes up to the top which means an intervention of it. Furthermore, the toxicity of the chemical used increases as we move further up the IPM pyramid.

All the treatments or approaches have certain disadvantages. This was the crux of the matter to do this project. Our proposed method should leave out all the drawbacks and only reduce the Varroa mites' population without causing any negative effects

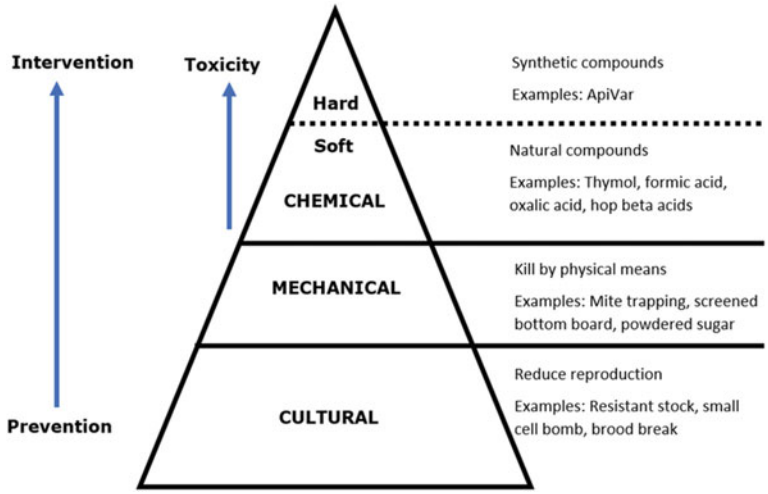


Fig. 1 The integrated pest management pyramid for Varroa control [18]

on the honey bees' colonies and without leaving any harmful residuals in the bees' products which also can be dangerous for human consumption.

2.1 Cultural or Biological Approaches

The cultural or biological control of the Varroa mite means that there is no use of chemicals to reduce the mite population. It uses the bee's biology to achieve this. The aim is to form a mite resistance in the bee colony due to higher grooming and hygiene activities or to lower the attractiveness of the brood to the mites. Furthermore, shorter post-capping methods and low mite fecundity factors can help to get a better resistance against the parasite. It is also possible to treat the beehive and the Varroa mites with temperature. This is because the Varroa mites are more susceptible to temperatures above 34 °C than the honey bees' larvae and pupae.

So, if the temperature reaches a value higher than 8 °C, the reproduction of the mite would be rendered ineffective. But the effort to reach a reduced mite population and at the same time, maintain a healthy bee colony is rather difficult compared to other solutions [3].

2.2 Mechanical Approaches

Another method to reduce Varroa is a mechanical approach with mite trapping, screened bottom board or powdered sugar. The mite trapping means the removal of

the drone brood. This is due to the fact that there is a higher success for the mite's reproduction as compared to the worker's brood.

Additionally, the drone brood attracts the mites for a longer period of time than the worker's brood. The parasite deposits its eggs into the brood's cells which contain the young bee larvae just before the cells are closed. If they are caught in the cells the bee keeper can remove this brood from the beehive to kill the mites. But this method will also kill the bee larvae. Because of this and the fact that not all of the mites can be killed with this method, another approach has to be devised [6].

A second mechanical concept is the so-called Muller Brett. Figure 2 shows graphically how Muller Bret works. The Muller Brett has two grids in a wooden frame. The upper one has a mesh width of 2 mm so that the Varroa can go through it. The lower grid has a mesh of 0.2 mm which do not enable the mite to pass. Additionally, an entrance for the bees to enter or exit has to be in the frame. To use this system, it has to be located between the upper and lower part of a beehive, as indicated by the red segment in Fig. 2. When the larvae come out of the bee's eggs, it takes six days until they grow in the closed honeycomb. Because the Varroa mite knows this through the scent of the larvae, it goes into the honeycomb cells, too. Once this is done, the bee keeper takes the closed brood and put it in the green segment in Fig. 2. This means that the larvae in this area are used to attract the mites. At the same time

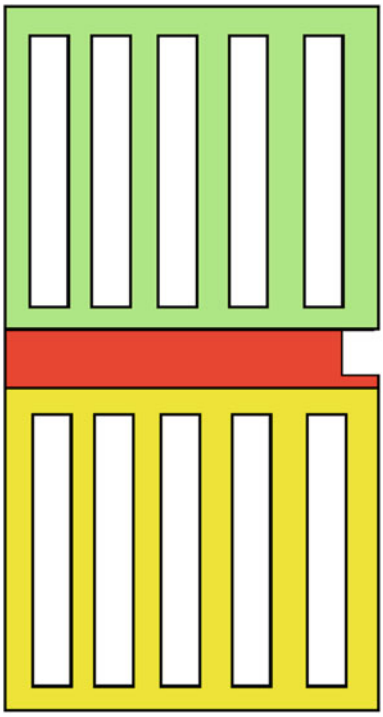


Fig. 2 Schematic of the function of the Muller Brett

the bee's queen is moved into the yellow area where she lays eggs and the larvae grow up. The mite is attracted by the scent of this young larvae and want to enter the yellow area. But because of the fine 0.2 mm mesh net, they cannot pass and will die of starvation. The advantage of this method is that no chemical substances are used for this case. But a certain amount of bee larvae has to be sacrificed to achieve this [17].

2.3 Chemical Approaches

The chemical treatment can be classified into the soft and hard chemical applications. For the soft chemical treatment oxalic acid is often used to kill the Varroa mites. The acid can be vaporized or put into a solution of sugar syrup and pure OA dihydrate. The problem of this treatment is that its effectiveness depends on the season where this is done. So, this means that it is necessary to know before hand in which season the treatment is effective [2]. Furthermore, it increases the larval mortality and by overusing this acid it can decrease the longevity and activity of worker bees [18].

For hard chemical treatment Amitraz can be used. The synthetic miticides are effective in eliminating up to 95% of the mite population. But the big disadvantage is the toxic effect of it. It influences the bees massively and make them more susceptible to Nosema disease. Furthermore, residuals of the toxic chemical can also be found in the wax or honey, which can be harmful for consumers (ibid.).

3 Hardware

To get an overview of the hardware components which are used in this project one can look at Fig. 3. The system is powered by a voltage source with 5 V. The heating element for the vaporizer is recommended to be supplied with a 12 V source.

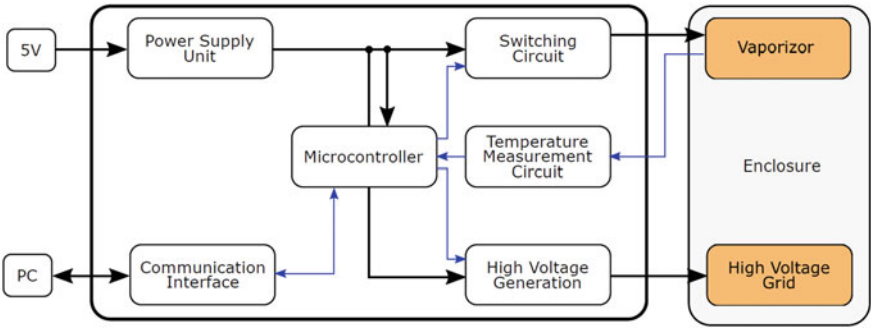


Fig. 3 Block diagram of the systems hardware components

Table 1 Heating element power measurement

Voltage (V)	4	5	6	7	8	9	10	11	12
Current (A)	1.17	1.46	1.75	2.03	2.30	2.58	2.84	3.15	3.37
Power (W)	4.68	7.3	10.5	14.2	18.4	23.2	28.4	34.7	40.4

Table 1 shows the electrical power which is consumed by the heating element at different voltages. It shows that the element has an electrical power between 5.68 W (at 4 V) and 40.4 W (at 12 V). To reduce the current which has to go from the external power supply through the PCB to the heating element one can reduce the voltage of the voltage source for the heater. As one can see in Table 1, the power consumption at 5 V is 7.3 W and therefore a current of 1.46 A is necessary to power the heating element at 5 V. Because of this reason, the supply voltage for the heating element was set to 5 V, which means an external power supply with 5 V is needed for this system.

As shown in Fig. 3, the core of the system is the microcontroller. All data or information are collected and stored in the microcontroller. There is a communication interface which allows the user to interact with the system. To vaporize the pheromones a vaporizer block is installed in an enclosure. For temperature control of the vaporization a switching circuit and a temperature measurement circuit are used. Furthermore, a block for the high voltage generation is located on the systems PCB. It is connected to the high voltage grid which is embedded in the side wall of the vaporizer enclosure.

3.1 Power Supply

The whole system is powered by an external power supply unit. Measurements at laboratory shown that the heating elements' current consumption is at 1.46 A, the current consumption of the high voltage circuit is at 200 mA maximum and the rest of the PCB's circuits need 50 mA in total. This means that an external power supply with an output voltage and current of 5 V and 2 A respectively is recommended for this system.

To prevent the PCB against reverse polarity the circuit shown in Fig. 4 is used. The MOSFET T_1 is a p-channel MOS Transistor and in combination with the zener diode D_1 and resistor R_5. If a positive voltage is connected to the input of the system the transistor switches the positive voltage to the system. In this case, an indicator LED shows that the systems' 5 V are connected. In case of a negative voltage at the input, the transistor blocks this voltage and the circuits.

The 5 V external voltage source is used for the heating element. Other subcircuits like the microcontroller, the operational amplifiers (Op-amp) for temperature measurement and the circuit for the high voltage generation use 3.3 V and 2.5 V to 3 V, respectively. Therefore, two voltage control circuits are necessary. Figure 5

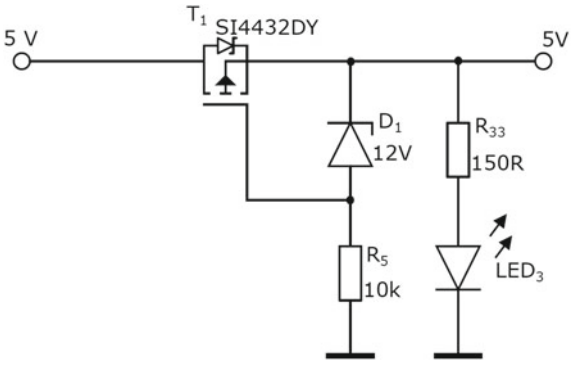


Fig. 4 Reverse polarity protection of the power line

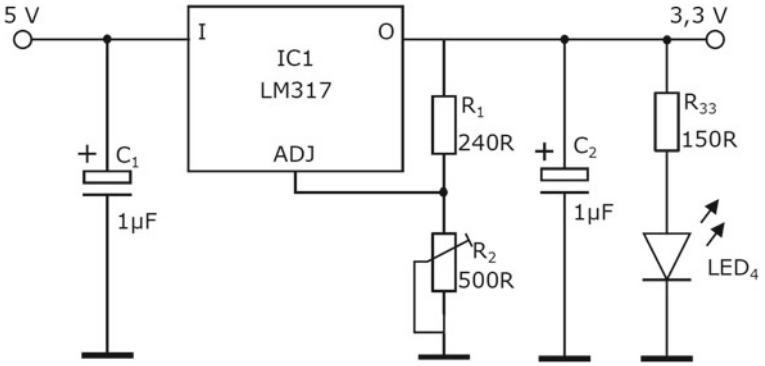


Fig. 5 Voltage regulator for 3.3 V output

shows the voltage regulator circuit for the 3.3 V power line. The use of integrated circuit (IC) LM317 is a low dropout regulator (LDO) and is used to generate an output voltage of 3.3 V from 5 V.

The LM317 is an adjustable voltage regulator which means the output voltage can be adjusted with a voltage divider, where value of the upper resistor (R_1) is fixed and the lower resistor (R_2) is a potentiometer. For stability and filtering noise a tantalum capacitor across input and the output line are recommended in the datasheet Texas Instruments 2020 for the LM317 voltage regulator. To indicate the 3.3 V power line a LED is used. As this circuit is designed in such a way that the output voltage can be adjusted, it is also possible to use it for other voltages. Therefore, the circuit from Fig. 5 is duplicated and used as voltage source with 2.5 V to 3 V output for the high voltage generation.

As one can see all VDD ports, the VBAT port and VDDA port are connected to the 3.3 V power line. To filter the 3.3 V line directly at the microcontroller a 100 µF capacitor is placed between the power line and ground. Additionally, a 4.7 µF

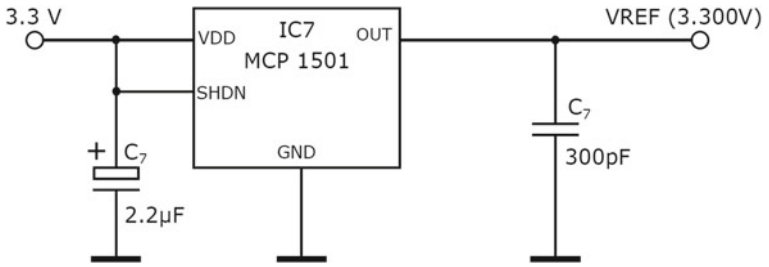


Fig. 6 Voltage reference for the ADC of the microcontroller

capacitor is recommended to be inserted parallel to the power line near the µC. The V_{SS} ports are all connected to the ground plane of the PCB.

If one wishes to use the analog-to-digital converter (ADC) of the microcontroller, an external voltage reference is recommended, as shown in Fig. 6. The output of the voltage reference is connected to the V_{REF} pin of the microcontroller. This is mandatory for obtaining accurate results from the ADC.

3.2 Microcontroller

A microcontroller is nothing more than a microchip which performs mathematical computations [12, 13, 20–22]. It is regarded as the 'brain' of a system, which in this case, is a robotic system. This project requires a microcontroller with digital input/output (I/O) ports, an analog-to-digital converter (ADC) and a serial communication interface. For this project, the STM32F407 in the LQFP100 package is used.

3.3 Communication Interface

To use the robotic Varroa trap system, an interface between the user and the system is mandatory. In this project, the user can interact with the system with a terminal program on his PC which is connected to the USB jack on the PCB. A simple way to realize this is a USB to UART converter. It converts the signals on the USB lines into serial messages and vice versa. The serial data lines are routed to the UART interface of the microcontroller. Figure 7 shows the USB to UART converter. On the left side is the USB jack with its four pins.

On pin 1 of the jack, the USB voltage of 5 V is applied. The 5 V USB line is protected with a 500 mA re-settable fuse and a reverse polarity protection. Furthermore, the data lines USB+ and USB− are connected to the FT232RL integrated circuit. And the ground pin is connected to the ground plane of the PCB. As one can

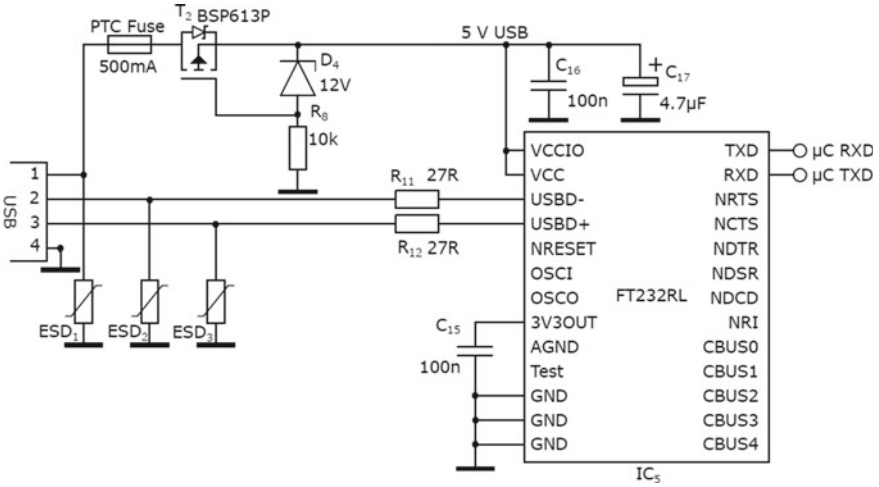


Fig. 7 USB to Serial/UART converter

see three electrostatic discharge (ESD) suppressors are used to protect the circuit from ESD pulses. This has to be performed because at the USB jack it is possible to touch the systems PCB and therefore, an electrostatic discharge can occur. To exchange data between the FT232RL IC and the microcontroller the UART interface with Receive Data (RXD) and Transmit Data (TXD) is used.

3.4 Switching Circuit

For switching the heating elements and the high voltage generation supply voltage on and off a switching circuit is necessary. This can be realized as shown in Fig. 8. Here, an n-channel MOSFET in configuration as a low side switch performs this task. The gate of the transistor is connected to the general-purpose input output (GPIO) pin of the microcontroller (µC).

If the GPIO output is set to logical LOW level, the transistor blocks and the heating element is not powered. In case of a logical HIGH at the output the MOSFET gets low resistive between source and drain and the current can flow from the 5 V power line through the heating element to the ground plane. For stability of the 5 V power line a 100 µF capacitor is placed near the transistor. To find out if the heating element is switched on or off a status indicator LED is used. A second but equal circuit is used to switch on and off the supply voltage of the high voltage generation. There is also an indicator LED to show if the high voltage generation is activated or powered off.

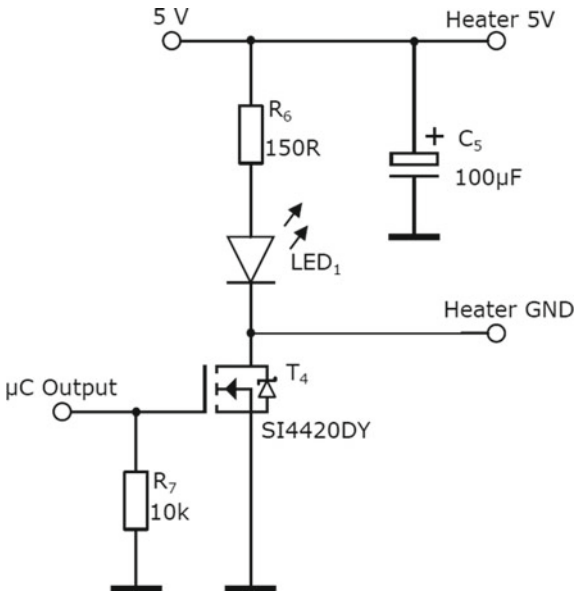


Fig. 8 Switching circuit for the heating element

3.5 High Voltage Generation

To generate a voltage of around 1000–2000 V for the high voltage grid, a blocking oscillator was combined with a Villard Cascade. The blocking oscillator is to generate an alternating voltage out of a direct voltage. This alternating voltage feeds the Villard Cascade through a transformer module. Figure 9 shows the circuit for the high voltage generation. The combination of transistor T_1, L_1 and L_2 forms a circuit which oscillates. Coils L_1 and L_3 are used as a transformer, where coil L_1 is the primary and coil L_3 is the secondary. This means that the input of the Villard Cascade gets the transformed alternating current to generate a high DC voltage from it. When the high voltage generation circuit is switched on, the blocking oscillator starts to oscillate with an amplitude of 3 V, depending on the input voltage. This alternating voltage is transformed by L_1 and L_3 to the input of the Villard Cascade. Equation 1 shows the transforming relations of an ideal transformer. To get the voltage on the secondary side of the transformer, Equation 2 is used. As one can see, the voltage on the secondary side is ideally 483.7 V. This voltage gets rectified and doubled due to the fact that this Villard Cascade is a voltage multiplier with the factor of 2. Therefore, at the output of the high voltage circuit a voltage of 967.5 V is provided. Due to the tolerances of the components and the fact that the transformer is real and not ideal, the measured voltage can differ from the theoretical value. For this case, the input voltage of the high voltage generation is adjustable and so it is possible to adjust the high voltage to the desired value.

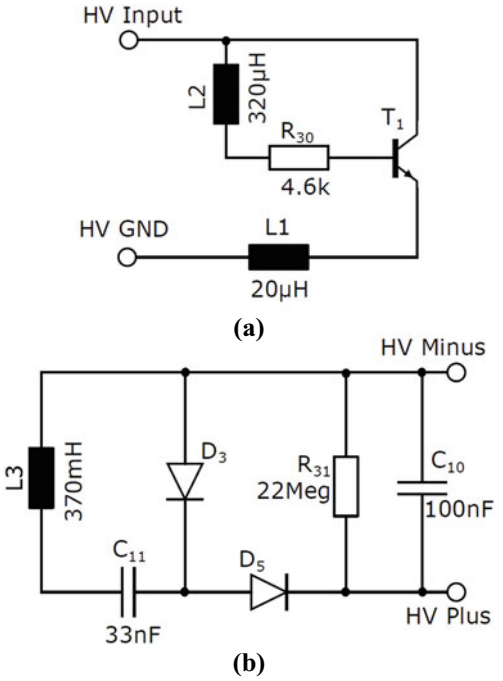


Fig. 9 A high voltage generation circuit which consists of **a** a blocking oscillator and **b** Villard Cascade

$$u = \sqrt{\frac{L_3}{L_1}} = \frac{U_{L_3}}{U_{L_1}} \tag{1}$$

$$U_{L_3} = U_{L_1} \times \sqrt{\frac{L_3}{L_1}} = 3\,\mathrm{V} \times \sqrt{\frac{390\,\mathrm{mH}}{15\,\mu H}} = 838.7\,\mathrm{V} \tag{2}$$

3.6 Temperature Measurement

For this project, a temperature sensor called'Thermistor with negative temperature coefficient (NTC)' is used. To realize the measurement, the circuit in Fig. 10 is implemented. The NTC resistor is externally installed in the heating block and connected to the measurement circuit with two wires. The voltage divider contains of the R_{16}, R_{17} and R_{NTC} resistors. The R_{17} resistor is used to reduce the current. A parallel capacitor

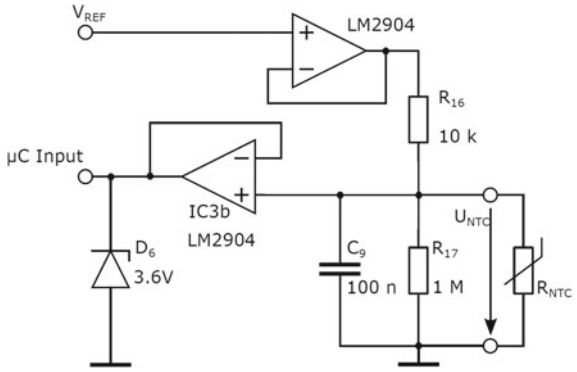


Fig. 10 Circuit for NTC temperature measurement

with 100 nF is placed for voltage stability. This voltage divider is supplied by the voltage V_{REF} from the external voltage reference through an operational amplifier (op-amp) in configuration as an impedance converter. A second op-amp is used as an impedance converter between the voltage divider and the input pin of the microcontroller which is internally connected to the ADC. The voltage divider is designed in such a way that the maximum voltage which can reach the microcontroller pin is V_{REF} (3 V) and therefore recommended to protect the microcontroller. In the datasheet of the NTC resistor.

The voltage at the NTC resistor can be computed with Eq. 3:

$$V_{NTC} = \frac{R_{tot} \times V_{REF}}{R_{REF} + R_{tot}} \tag{3}$$

where R_{tot} is the parallel combination of R_{NTC} and R_{17}.

Now it is possible to convert the voltage VNTC with the help of the analog-to-digital converter inside the microcontroller to a digital value. The STM32F4 µC has a resolution of 12 bit which means that there are digital values between 0 and 4095 (i.e. the resolution is 4095). To obtain the digital value V_{ADC}, Eq. 4 below is applied:

$$V_{ADC} = V_{NTC} \times \frac{\text{Resolution}}{V_{REF}} = V_{NTC} \times \frac{4095}{3} \tag{4}$$

Equation (4) allows the R_{NTC}, V_{NTC} and the ADC values to be tabulated in a lookup table for every temperature.

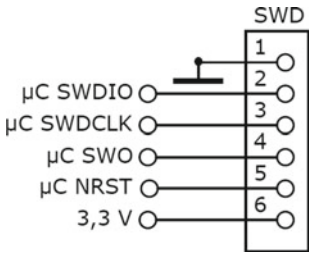


Fig. 11 SWD interface for the microcontroller

3.7 Miscellaneous Circuits

In this section, three circuits which are necessary for the functionality of the robotic system are described. The first is the so-called Serial Wire Debug (SWD) interface. It is necessary for programming and debugging the microcontroller. Figure 11 shows the SWD connector which is collocated with the microcontroller on the printed circuit board (PCB).

To reset the microcontroller if necessary or to start the system, a 'reset' button and a 'user' button are installed. As one can see in Fig. 12, the connection to the microcontroller is always on logical HIGH level because of a pull-up resistor. If the either the 'reset' or 'user' button is pressed, the logic level on the microcontroller input pin is changed to a logic LOW and therefore the system is reset (when the 'reset' button is pressed) or started (when the 'user' button is pressed).

Additionally, an interface for display is also implemented on the PCB. With this interface, it is possible to display an 8-bit data. Figure 13 shows the display connector interface.

To complete the system, a rotary knob interface is built. Figure 14 depicts the hardware connector for the encoder.

3.8 Vaporizer

For the vaporizer, a hotend from a 3D printer is modified and used. The heating element is a ceramic heater with a maximum electrical power of 40 W. For this project, the heating power is not as high as compared with the hotend system of a 3D printer. Therefore, the heating element is powered with 5 V which equivalent to an electrical power of 7.3 W. For temperature measurement, an NTC is used. As shown in Fig. 15, the heating element and the NTC resistor are embedded and fixed in a block made out of aluminium. Therefore, two holes are drilled into the block. Additionally, a blind hole on the top surface for the liquid pheromones is attached.

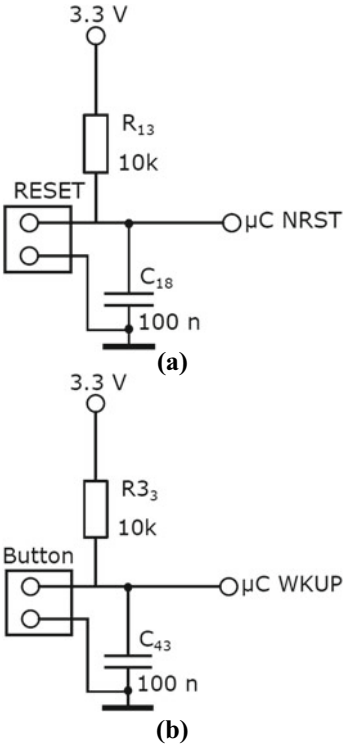


Fig. 12 The (**a**) reset button and (**b**) user button circuits

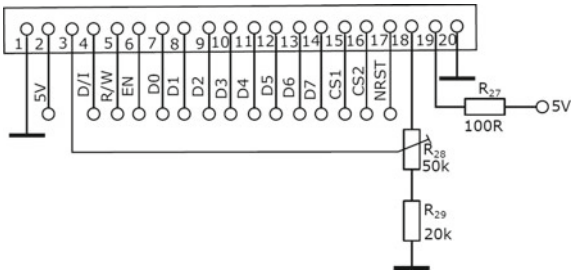


Fig. 13 Display connector for graphic display

3.9 Enclosure and High Voltage Grid

To kill the Varroa mite a high voltage grid is used. This grid is installed in an enclosure which is produced in the 3D printing process. The grid is made out of four layers of copper mesh. Three layers are for killing the mites, the outer one is to prevent

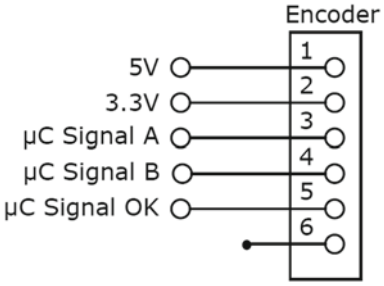


Fig. 14 Encoder interface for rotary knob

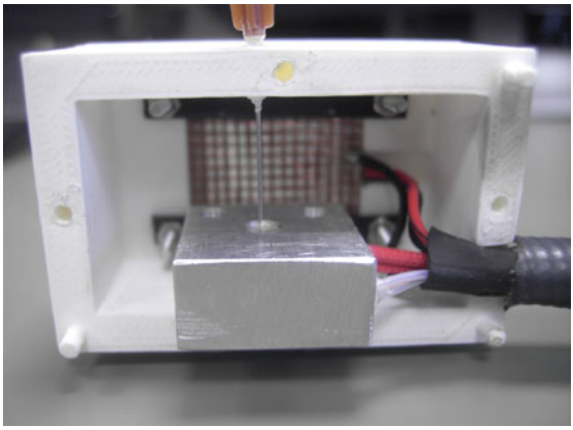


Fig. 15 The vaporizer block installed with a heating element and an NTC

the bees from being killed. Between every layer of copper mesh is an isolator on the outside so that no mesh layer gets in contact with another layer.

The middle layer of the three 'Working Meshes' is connected to the positive terminal of the high voltage source, the two covering layers are connected to the negative terminal of the high voltage source. If a mite enters the enclosure through the high voltage grid and touches two of the 'Working Meshes' it will be killed by the high voltage. Figure 16 shows the printed enclosure with the installed high voltage grid. On the top surface of the enclosure a needle for a syringe is installed. It is placed right above the blind hole of the heating block to put the liquid pheromones on the vaporizer. This can be obtained from Fig. 15. The designed and equipped PCB is also embedded in an enclosure. Figure 17 shows the PCB inside the enclosure.

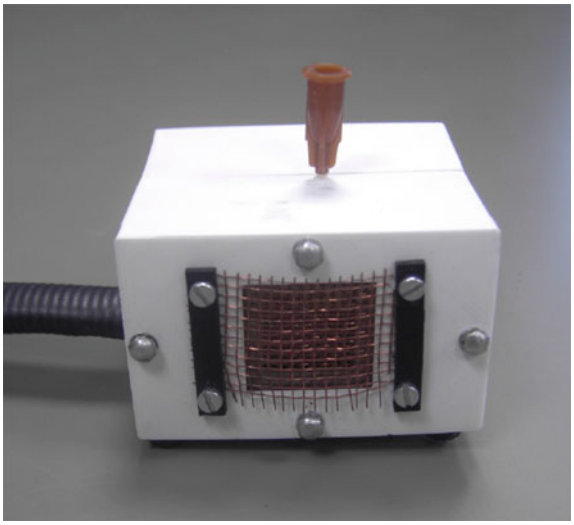


Fig. 16 Enclosure with integrated high voltage grid

Fig. 17 PCB embedded in a metal enclosure

4 Software

The software part of this project can be summarized in a block diagram. Figure 18 shows this diagram. As one can see, it is divided into five groups. The core of the project software is the main function. It coordinates every task which has to be performed. One part is dedicated for the communication with the PC or the terminal. The other parts are responsible for temperature control, user button and enabling respectively disabling the high voltage circuit.

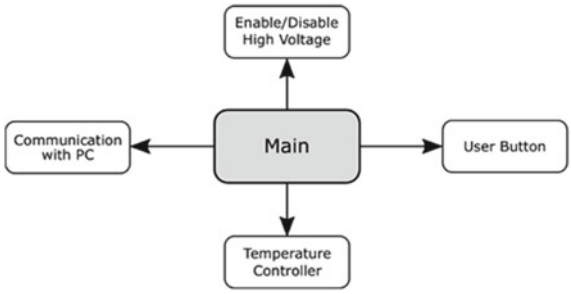


Fig. 18 Block diagram for the software

4.1 The IDE Coocox and Standard Peripheral

To implement software on a microcontroller, an Integrated Development Environment (IDE) is recommended. In this project, the license and charge free IDE 'Coocox' was used for coding, debugging and flashing the microcontroller. The programmes were written using the C programming language. The IDE includes a compiler that translates the C code into a machine code running on the microcontroller. For a readable and an intuitive coding, the 'Standard Peripheral Drivers' from ST Microelectronics are used. They have to be included in the project tree for usage.

4.2 The Main Function

The main function of the software is displayed in Fig. 19. When the microcontroller is powered, all necessary components are initialized. After the initialization, the microcontroller will wait for an input signal from the 'user' button. When the 'user' button is pressed, the function 'Start Menu' is called. After returning from this function, the main function again waits for the 'user' button status. This created an infinite loop, so that the microcontroller does not come across an undefined state.

4.3 The Initialization Functions

In this project, the GPIOs, the UART interface and the ADC of the microcontroller are used. Therefore, every unit has to be initialized in order to obtain correct functionality. The first step is to configure the clock of the microcontroller which is done by the 'System Init' function. It is a part of the standard peripheral drivers to set the external crystal as source for the clock generation.

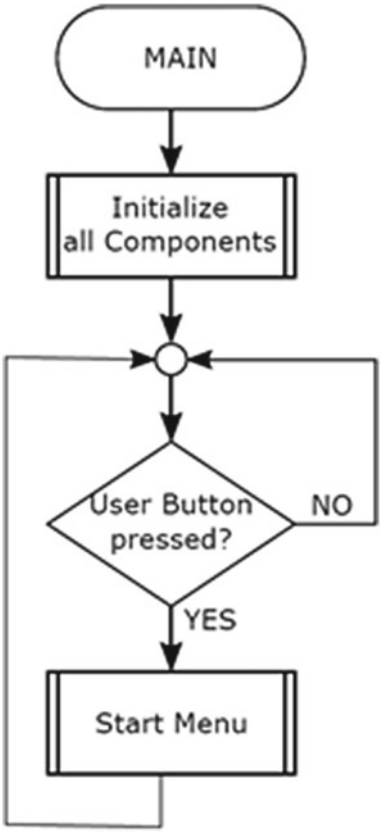


Fig. 19 Flow chart of the main function

After this, the UART interface will be configured. Listing 1 shows how the ADC is configured for the desired functionality of temperature measurement. First of all, the clock for the ADC has to be enabled in line 17. Then, the resolution is set to 12 bit and the scan mode disabled because only one channel is used. Furthermore, the continuous conversion has to be enabled for continuous measurement. The trigger for starting the measurement is done by software and therefore, the external trigger is disabled. Line 28 configures the converted 12-bit data to be aligned on the right side of the 16-bit data register. These are the most important configurations of the ADC. For temperature control, the so called analog watchdog (AWD) interrupt has to be enabled. This is done from line 49 to 66. After this configuration, the ADC is enabled in line 69 and it is ready to be used in the application.

Listing 1. ADC Configuration

```
// ***********************************************************************
// Configuration of ADC1
// ADC Resolution: 12 Bit
// ADC Mode: Continuous Mode
// ADC Interrupts: Analog Watchdog Interrupt
// ***********************************************************************
void ADC1_Configuration(void)
{
      //Init of AD Struct
      ADC_InitTypeDef ADC_InitTypeStructure;
      //Init AD Common Structure
      AD_CommonInitTypeDef ADC_CommonStructure;
      //Initialize NVIC Struct
      NVIC_InitTypeDef NVIC_InitStructure;

      // ADC1 CLK EN
      RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

      //Res 12 Bit
      ADC_InitTypeStructure.ADC_Resolution = ADC_Resolution_12b;
      //Disable Scan Mode (only 1 Channel is needed)
      ADC_InitTypeStructure.ADC_ScanConvMode = DISABLE;
      //Cont Mode
      ADC_InitTypeStructure.ADC_ContinuousConvMode = ENABLE;
      //No Ext Trig
      ADC_InitTypeStructure.ADC_ExternalTrigConvEdge =
          ADC_ExternalTrigConvEdge_None;
      //Align Data Right (12 Bit in 16Bit Reg)
      ADC_InitTypeStructure.ADC_DataAlign = ADC_DataAlign_Right;
      //Number of Convs set to 1
      ADC_InitTypeStructure.ADC_NbrOfConversion=1;

      //Only AD Converter 1 is used
      AD_CommonStructure.ADC_Mode = ADC_Mode_Independent;
```

Listing 1. (Continued) ADC Configuration

```
34        //AD Converter PSC
35        AD_CommonStructure.ADC_Prescaler = ADC_Prescaler_Div8;
36        //DMA Mode not enable
37        AD_CommonStructure.ADC_DMAAccessMode =
              ↪ ADC_DMAAccessMode_Disabled;
38        //Delay of two Samples
39        AD_CommonStructure.ADC_TwoSamplingDelay =
              ↪ ADC_TwoSamplingDelay_5Cycles;
40        //ADC CH for Measurement is CH 11
41        AD_RegularChannelCfg(ADC1, ADC_Channel_11, 1,
              ↪ AD_SampleTime_480Cycles);
42
43        //Set all Configs for ADC 1
44        ADC_Init(ADC1, &ADC_InitTypeStructure);
45        AD_CommonInit(&ADC_CommonStructure);
46        //Set INT
47        NVIC_InitStructure.NVIC_IRQChannel = ADC_IRQn;
48        //Set INT Prio
49        NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 1;
50        NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
51        //Enable INT Channel for ADC
52        NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
53        //Init of all ADC INT Cfg
54        NVIC_Init(&NVIC_InitStructure);
55
56        //Set default HTR and LTR
57        HighThreshold = ADC_Value_35deg + 1;
58        LowThreshold = ADC_Value_35deg - 1;
59
60        ADC_AnalogWatchdogThresholdsConfig(ADC1, HighThreshold,
              ↪ LowThreshold);
61        //Set CH 11 Analog Watchdog INT
62        ADC_AnalogWatchdogSingleChannelConfig(ADC1, ADC_Channel_11);
63        ADC_AnalogWatchdogCmd(ADC1, ADC_AnalogWatchdog_SingleRegEnable
              ↪ );
64
65        //Enable ADC1
66        AD_Cmd(ADC1, ENABLE);
67        //Start
68        ADC_SoftwareStartConv(ADC1);
69    }
```

The next step is to configure the UART interface for communication. The steps are shown in Listing 2. The first step is to enable the clock source for the GPIO pins and the UART peripheral. Next, the UART interface is mapped on the two GPIO pins GPIOA pin 9 and GPIOA pin 10 as shown in line 19 and 20. Then the UART interface is specified in line 38–51. The baudrate is set to 9600 Baud, flow control is disabled and the mode is set to receive and transmission. Furthermore, no parity bits, one stop bit and an 8-bit data field are configured. The last step is to enable the UART interface in line 52.

To use the button and to switch the heating element and the high voltage circuit GPIOs are used. Listing 3 contains the source code for initialization of the GPIO for the user button from line 6–24 and the init for switching the heating element and the high voltage source from line 28–60. For all pins the clock source has to be enabled first. After this the GPIO pin for the user button is set as an input and no internal pull resistor is used. The two GPIO pins for switching are set as outputs without internal pull resistors. Additionally, these two pins are configured as a push pull output. After these steps, the GPIO pins are configured for the desired functionality.

4.4 Communication with Computer

For communication or interaction of the system with the user a UART interface and a terminal application on the user's computer is implemented. Figure 20 illustrates the menu structure and communication to the system.

When the user button is pressed in the main loop, the routine 'Start Menu' is called and the main menu is displayed in the terminal. As long as the exit flag is not set, three options for the user are available and can be choose by entering different keys on the keyboard. When the user enters a '1' it is possible to set the vaporization temperature. The subroutine 'Get Entered Temperature' is waiting until the user enters a desired temperature value. A query if this value is allowed is performed so only a certain range of temperature is permitted.

The second option which means the user has to enter '2', starts the vaporization application. In this case the high voltage source is enabled. Furthermore, the temperature control system is switched on by enabling the AWD interrupt. More details about the AWD interrupt can be found in the next section of the temperature control. The application is running as long as the user does not abort it by entering'0'. After every menu selection it is possible to return to the main menu by pressing any key. This is only available for the first and second menu option. The third option is to leave the menu by entering a '3' on the keyboard. Now the exit flag is set and the program returns to the main routine.

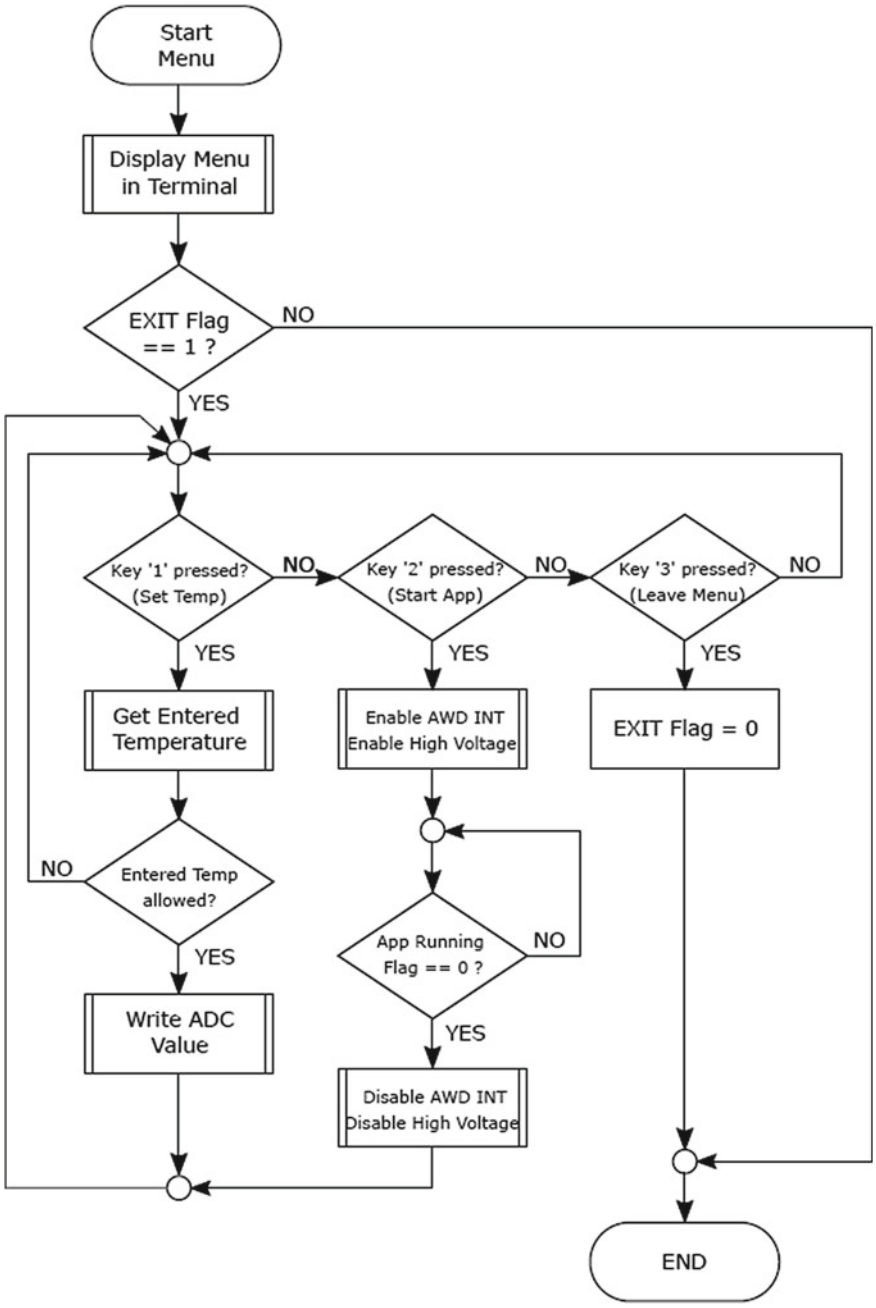


Fig. 20 Flow chart of how the system communicates with the computer

Listing 2. UART Configuration

```
// ***********************************************************************
// Configuration of UART1 Interface
// Baudrate: 9600
// UART Mode: RX and TX
// Wordlength: 8 Bit
// Used Pins: PA9, PA10
// ***********************************************************************
void USART_init(void)
{
   // Init General Purpose I/O Struct
   GPIO_InitTypeDef  GPIO_InitStruct;

   // Clk EN for GPIOA
   RCC_AHP1PeriphClockCmd(RCC_AHP1Periph_GPIOA, ENABLE);
   // Clk UART1
   RCC_AHB2PeriphClockCmd(RCC_AHB2Periph_USART1, ENABLE);

   // Map GPIOA Pin9 as UART1_RX and Pin10 as UART1_TX
   GPIO_PinAFConfig(GPIOA, GPIO_PinSource9, GPIO_AF_USART1);
   GPIO_PinAFConfig(GPIOA, GPIO_PinSource10, GPIO_AF_USART1);

   //Use alternate Function for GPIOA Pin 9 and Pin 10
   GPIO_InitStruct.GPIO_Pin = GPIO_Pin_9 | GPIO_Pin_10;
   GPIO_InitStruct.GPIO_Mode = GPIO_Mode_AF;

   // Configure GPIOA Pin 9 and Pin10 with no internal Pull Resistor
      ↪ and 50 Mhz Speed
   GPIO_InitStruct.GPIO_PuPd = GPIO_PuPd_NoPull;
   GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;

   //Set Configs of GPIOA Pin9 and Pin10
   GPIO_Init(GPIOA, &GPIO_InitStruct);

   // Ini Strct
   USART_InitTypeDef USART_InitStruct;

   // Set Baudrate to 9600 Baud
   USART_InitStruct.USART_BaudRate = 9600;
   // No Hardware Ctrl
   USART_InitStruct.USART_HardwareFlowControl =
      ↪ USART_HardwareFlowControl_None;
```

Listing 2. (Continued) UART Configuration

```
    // Set UART Mode RX, TX
    USART_InitStruct.USART_Mode = USART_Mode_Tx | USART_Mode_Rx;
    // No Par Bit used
    USART_InitStruct.USART_Parity = USART_Parity_No;
    // Stop Bit is 1 Bit
    USART_InitStruct.USART_StopBits = USART_StopBits_1;
    // UART Wordlength is set to 8 Bit
    USART_InitStruct.USART_WordLength = USART_WordLength_8b;
    //Set all Configs
    USART_Init(USART1, &USART_InitStruct);
    // Run it
    USART_Cmd(USART1, ENABLE);
}
```

Listing 3. Button Configuration

```
// ***************************************************************************
// Configuration of used GPIO Pins for User Button
// Used GPIOs: GPIOA
// Used Pins: GPIOA Pin 0
// ***************************************************************************
void Button_Init()
{
    //Ini
    GPIO_InitTypeDef  GPIOx_InitStructure;
     //Clk EN for GPIOA
     RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);
     //Use GPIOA Pin 0
     GPIOx_InitStructure.GPIO_Pin = GPIO_Pin_0;
    //In
     GPIOx_InitStructure.GPIO_Mode = GPIO_Mode_IN;
     //No Pull
     GPIOx_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
    //50 MHz Speed
     GPIOx_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    //Set all Cfgs of Pin
     GPIOx_Init(GPIOA, &GPIO_InitStructure);
}

```

Listing 3. (Continued) Button Configuration

```
// ********************************************************************
// Configuration of used GPIO Pins
// Used GPIOs: GPIOB, GPIOC
// Used Pins: GPIOB Pin4, Pin5 and GPIOC Pin1
// ********************************************************************
void GPIO_Init()
{  // Init GPIO Struct
   GPIO_InitTypeDef  GPIOx_InitStructure;

   // Clk EN GPIOB
   RCC_AHP1PeriphClockCmd(RCC_AHP1Periph_GPIOB, ENABLE);
   // Clk
   RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);

   //Use GPIOB Pin 4 and Pin 5
   GPIOx_InitStructure.GPIO_Pin = GPIO_Pin_4 | GPIO_Pin_5;
   //Out
   GPIOx_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
   //Output Type PP
   GPIOx_InitStructure.GPIO_OType = GPIO_OType_PP;
   //No Pull
   GPIOx_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
   //50 MHz
   GPIOx_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
   //Set Cfg Pin4 Pin 5
   GPIOx_Init(GPIOB, &GPIO_InitStructure);
}
```

4.5 Temperature Control

The temperature control for the vaporizer is realized with an NTC which is able to indirectly measure the temperature. Furthermore, the switching circuit for the heating element is necessary. The implementation for the temperature control procedure is done with an ADC interrupt, especially an AWD interrupt and a related interrupt handler. For the AWD interrupt two threshold values where an interrupt is generated can be set.

These values correspond to the desired temperature the user wants for vaporization.

With this interrupt a two-point controller can be realized. As described in the previous section, the temperature for vaporization can be set by the user via the menu in the terminal. This value has to be converted into a digital ADC value and written into the registers for low threshold and high threshold of the ADC. Figure 21 shows the procedure to set the correct ADC threshold values for the temperature control system. A lookup table where the digital ADC values and the corresponding temperature values in degree Celsius is used. Listing 4 demonstrates how the lookup table is prepared. It is an array where the pairs of values are stored in the microcontroller and included as a header file. If the user enters his desired temperature value in degree Celsius the subroutine 'Write ADC Value' is called. A pointer to the lookup table array is generated. Now the algorithm is searching the entered temperature value in the table. If the value is found, the corresponding ADC value plus or minus one digit is set as the low and the high threshold value of the ADC which represents the hysteresis of the temperature control system. These values are responsible for triggering the ADC AWD interrupt. When the values are stored in the ADC registers the subroutine returns to the previous routine.

Figure 22 illustrates the ADC analog watchdog interrupt handler for the temperature control system. If an interrupt of the ADC is triggered, the running routine is suspended and the interrupt handler is called. Inside the handler function the control mechanism is implemented. The first step is to check whether the ADC AWD bit is set or not. This has to be done because there is only one ADC handler for all ADC interrupts and therefore, this request has to be done first. If the AWD bit is set, the AWD interrupt is disabled and the corresponding pending flag is cleared.

Now a comparison with the actual measured ADC value, which represents the actual temperature, and the thresholds is done. If the actual temperature is less than or equal to the lower threshold temperature the heating element is switched on. On the other hand, if the actual measured temperature is higher than or equal to the upper threshold temperature the heating element is switched off. Once this process is completed, the system will then return to the analog-to-digital conversion stage and the whole process is iteratively repeated. This control mechanism is performed as long as the user enables the vaporization procedure.

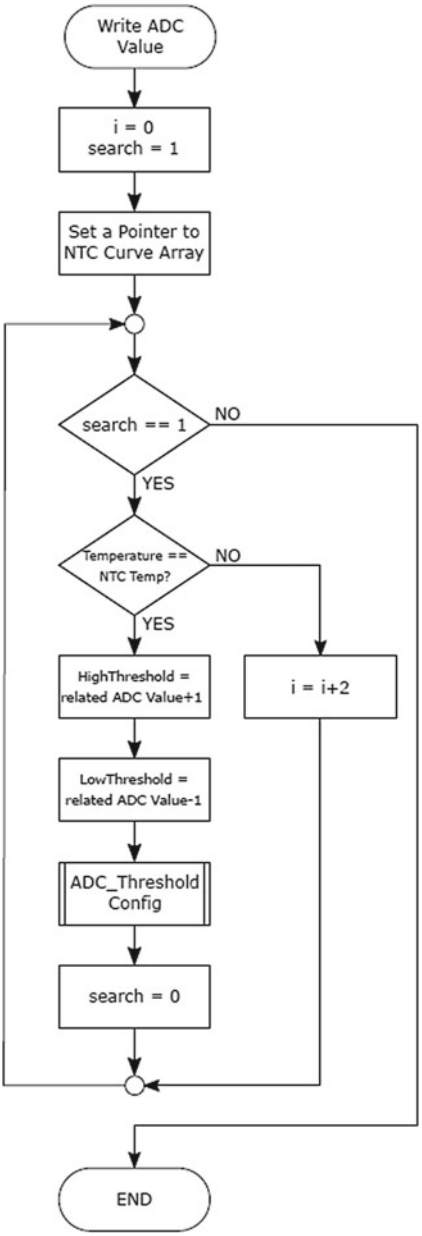


Fig. 21 Flow chart for setting the hysteresis upper and lower values

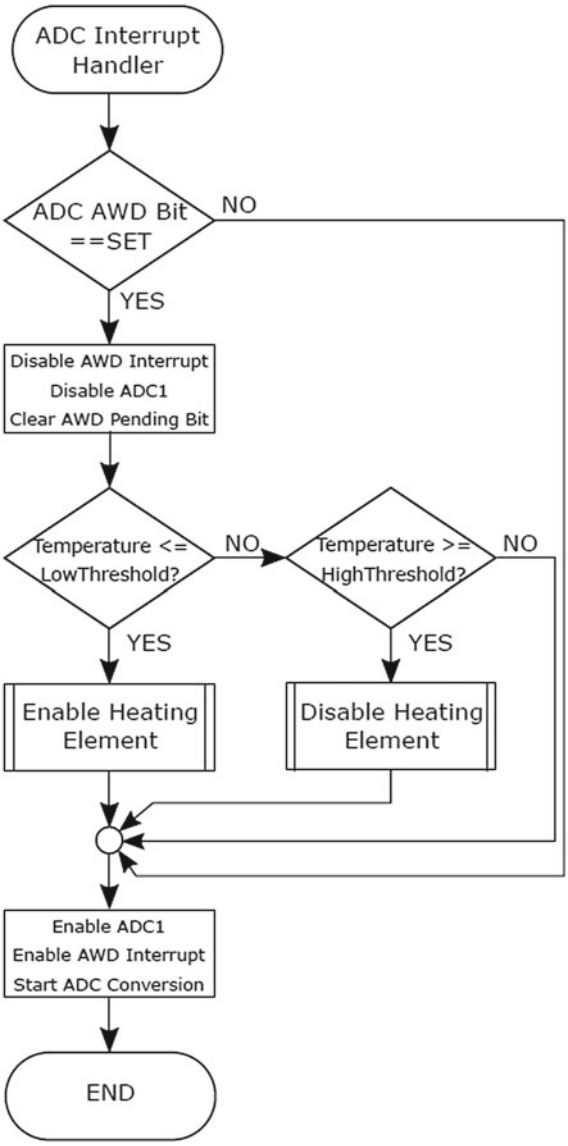


Fig. 22 Flow chart of the ADC handler for temperature control

Listing 4. NTC Lookup Table

```
// ***************************************************************************
// Project UTAR 2020
// Author: Johannes Meister
// Date: December 2020
// NTC Lookup Table
// ***************************************************************************
#ifndef _NTC_Curve_H
#define _NTC_Curve_H
static const uint32_t NTC_Curve[]={
      20,   2157,
      21,   2115,
      22,   2074,
      23,   2033,
      24,   1991,
      25,   1950,
      26,   1910,
      27,   1871,
      28,   1832,
      29,   1793,
      30,   1754,
      31,   1715,
      32,   1676,
      33,   1637,
      34,   1599,
      35,   1561,
      36,   1524,
      37,   1487,
      38,   1450,
      39,   1414,
      40,   1379,
      41,   1344,
      42,   1310,
      43,   1276,
};
#endif
```

4.6 The Complete System

The whole system consists of the combination of the hardware and the software parts from the previous sections. Figure 23 shows the completed enclosure with the installed high voltage grid, the vaporizer block and the syringe with needle for the fluid attractants. Furthermore, in Fig. 24 the completed system with the vaporizer and the control PCB in an enclosure are illustrated.

To use the system, the external power supply and a USB cable for connection with a PC is mandatory. When the system is powered up and the terminal program on the PC is connected to the USB cable, the user can push the black button on the enclosure. After pressing this button, the main menu as shown in Fig. 25. Now, it is possible to choose one of the three options below:

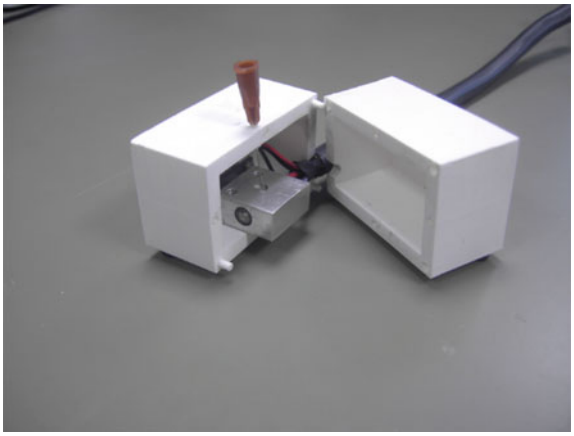


Fig. 23 The enclosure installed with high voltage grid, a vaporizer and a syringe

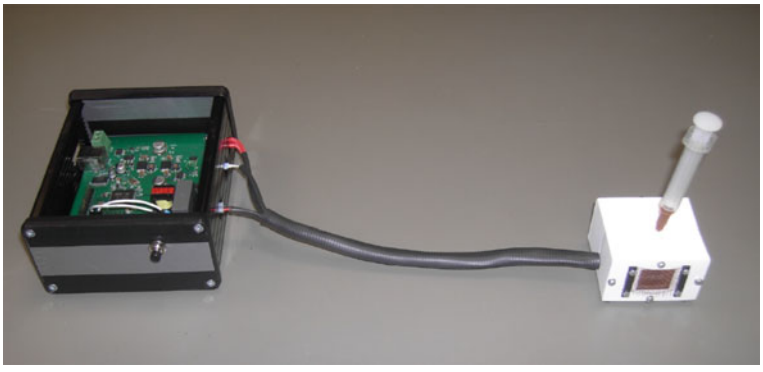


Fig. 24 The complete system

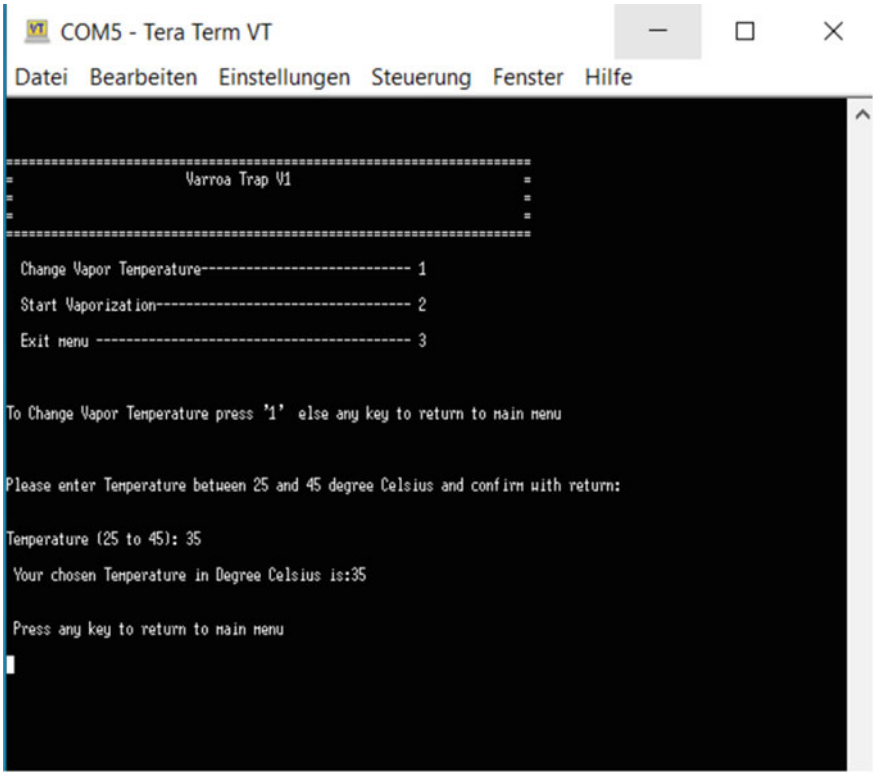

COM5 - Tera Term VT
Datei Bearbeiten Einstellungen Steuerung Fenster Hilfe
Varroa Trap V1
Change Vapor Temperature---------- 1
Start Vaporization---------- 2
Exit menu ---------- 3
To Change Vapor Temperature press '1' else any key to return to main menu
Please enter Temperature between 25 and 45 degree Celsius and confirm with return:
Temperature (25 to 45): 35
Your chosen Temperature in Degree Celsius is:35
Press any key to return to main menu

Fig. 25 The main menu shown in the terminal program

(i) Change vapor temperature
(ii) Start vaporization application
(iii) Exit the Main Menu

If the user has chosen item 2, the vaporization application starts and the high voltage grid is activated. By pressing '0' the vaporization is stopped and the user can return to the main menu as shown in Fig. 26.

5 Results

After designing the hardware, implementation of the software and combining them into a complete system, it had to be tested. Due to the fact that in winter time it is not recommended to disturb a bee colony, only tests of the fundamental functionality of this system have been done for verification. The first check has to be performed on the high voltage generation. With the help of the 'Voltcraft VC820' digital multimeter, the DC voltage at the output of the high voltage generator has been measured. It is

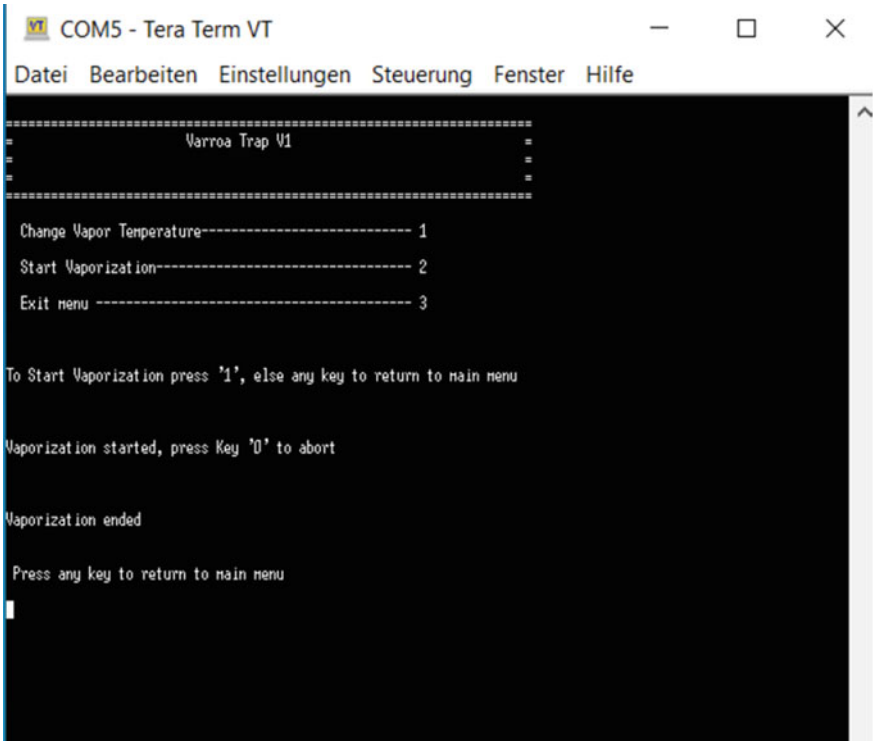

COM5 - Tera Term VT
Datei Bearbeiten Einstellungen Steuerung Fenster Hilfe
Varroa Trap V1
Change Vapor Temperature---------- 1
Start Vaporization---------- 2
Exit menu ---------- 3
To Start Vaporization press '1', else any key to return to main menu
Vaporization started, press Key '0' to abort
Vaporization ended
Press any key to return to main menu

Fig. 26 How vaporization is activated in the terminal program

possible to adjust this voltage with the LDO circuit of high voltage power supply. The DC voltage is adjusted to 1000 V. Furthermore, the high voltage grid should be tested. For this case, the HV grid is connected to the HV source and enabled. At a voltage of 1000 V, no flashover between the meshes can be detected. With the help of a small isolated screwdriver, it is possible to reach the high voltage meshes. When the screwdriver's tip touch two of the inner meshes, sparks can be detected. This means that the high voltage source and grid is proper working.

To verify the temperature control system, the 'DM620' temperature data logger which comes with an external temperature sensor is used. The temperature sensor is fixed on the surface of the aluminium heating block. Now, the data logger is started and the vaporizer is activated to work at a dedicated temperature. In this case a temperature of 30 °C is set to test the control algorithm. The limits are set to 29 °C and 31 °C because a deviation of ± 1 °C is allowed for the vaporization of the attractants. From the observation, it was found that the control system is able to hold the temperature of the vaporizer block in the given range of ± 1 °C around the nominal temperature. The results strongly suggest that the control system is able to function appropriately.

6 Conclusion

In this chapter, the construction of a robotic Varroa destructor trap with the usage of pheromones was demonstrated. The first step was to design an enclosure, an appropriate high voltage grid and a high voltage source. Furthermore, the development of a control system for vaporization the pheromones and finally implemented the prototype and tested it.

For the first step a small enclosure was designed and printed with a 3D printer. To kill the mites a high voltage grid made out of three working meshes which are stacked together is installed in the enclosure. An additional safety mesh above the high voltage grid to prevent the bees from touching the high voltage grid has been mounted.

Next, a microcontroller has been chosen to control the entire system. The STM32F407 was selected for this project. It is also used for implementing the temperature controller for vaporizing the attractants. Furthermore, a circuit for temperature measurement and a circuit for switching the heating elements power supply has been designed.

The last step includes the design and testing of the complete system. For the complete system are several hardware circuits and dedicated software parts used. First all hardware parts have been designed. This includes the reverse polarity protection, power supply 3.3 and 2.5 V, communication interface, high voltage generation, switching circuit, a circuit for temperature measurement and the circuit to power and to run the microcontroller. All these parts have been combined on a single PCB which is embedded in an enclosure. Furthermore, the vaporizer was installed in the enclosure with the high voltage grid and connected to the PCB. After designing the hardware, software implementations have been performed. The communication is done via a USB to UART interface and therefore the handling of the UART interface was part of the software. Additionally, the temperature controller has been implemented in software. This is done by measuring the actual temperature with an ADC and comparison with the configured temperature in an interrupt service routine.

Finally, all components of hardware and software have been tested as a complete system. The test includes a measurement of the high voltage and the check of the functionality of the high voltage grid. This test has been passed correctly. After this the temperature controller was verified by using an external temperature sensor and a data logger for tracing the temperature of the vaporizer block. This verification showed out that the temperature controller works as required which means the temperature can be hold within the limits of ± 1 °C around the nominal temperature. All in all, the system fulfills the requirements which were given and works in a proper way.

References

1. Abou-Shaara, H.F.: Continuous management of Varroa mite in honey bee, Apis mellifera, colonies (2014)
2. Akyol, E., Unalan, A.: Effect of oxalic acid treatment in different seasons on varroa (varroa destructor) population in honeybee colonies. Feb-Fresenius Environ. Bull. 3863 (2017)
3. Ayan, A., Tutun, H., Aldemir, O.S.: Control methods against Varroa mites. Int. J. Adv. Study Res. Work **11**(2), 19–23 (2019)
4. AlZu'bi, S., Hawashin, B., Mujahed, M., Jararweh, Y., Gupta, B.B.: An efficient employment of internet of multimedia things in smart and future agriculture. Multimedia Tools Appl **78**(20), 29581–29605 (2019)
5. Brockmann: Brockmann-Neue-Erfindungen-Hyperthermie". In: https://www.bienenundnatur.de/wp-content/uploads/2016/10/2015-01_ADIZ-db-IF_Brockmann-Neue-Erfindungen-Hyperthermie.pdf (2015). Accessed 8 Nov 2021
6. Devi, S.: Different methods for the management of Varroa mite (Varroa destructor) in honey bee colony (2019)
7. Elmisery, A.M., Sertovic, M., Gupta, B.B.: Cognitive privacy middleware for deep learning mashup in environmental IoT. IEEE Access **6**, 8029–8041 (2017)
8. Frazier, M.: Varroa mites. The Mid-Atlantic Apicultural Research and Extension Consortium. Publication **4**, 1–3 (2005)
9. Gupta, B.B., Tewari, A., Cvitić, I., Peraković, D., Chang, X.: Artificial intelligence empowered emails classifier for Internet of Things based systems in industry 4.0. Wireless Netw. 1–11 (2021)
10. Jánszky, B., Herrmann, F.: Auf Augenhöhe: Wissenstransfer zwischen Forschung und Praxis der ökologischen und nachhaltigen Land-und Lebensmittelwirtschaft (Teilschlussbericht 1 des BÖLW) (2019)
11. Jiang, F., Fu, Y., Gupta, B.B., Liang, Y., Rho, S., Lou, F., Meng, F., Tian, Z.: Deep learning based multi-channel intelligent attack detection for data security. IEEE Trans. Sust. Comput. **5**(2), 204–212 (2018)
12. Lee, W.T., Nisar, H., Malik, A.S., Yeap, K.H.: A brain computer interface for smart home control. In: 2013 IEEE International Symposium on Consumer Electronics (ISCE). IEEE, pp. 35–36 (2013)
13. Nisar, H., Khow, H.W., Yeap, K.H.: Brain-computer interface: controlling a robotic arm using facial expressions. Turk. J. Electr. Eng. Comput. Sci. **26**(2), 707–720 (2018)
14. Radetzki, T.: Forschungsergebnisse zur Oxalsäure Verdampfung aus dem Winterhalbjahr. http://www.apis.admin.ch/english/host/pdf/alternativ/York.OS0109Radetzki.pdf (2001)
15. Rosenkranz, P., Aumeier, P., Ziegelmann, B.: Biology and control of Varroa destructor. J. Invertebr. Pathol. **103**, S96–S119 (2010)
16. Sahoo, S.R., Gupta, B.B.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Appl. Soft Comput. **100**, 106983 (2021)
17. Schmid, C.: "Meine Erfahrung mit dem Muller–Brett". In: https://www.bienen-janisch.at/index.php?route=common/download/file&download_id=17 (2020). Accessed 8 Nov 2021
18. Underwood, R., López-Uribe, M.: Methods to Control Varroa Mites: An Integrated Pest Management approach. Pennsylvania State University Extension, University Park (2019)
19. Traynor, K.S., Mondet, F., de Miranda, J.R., Techer, M., Kowallik, V., Oddie, M.A., Chantawannakul, P., McAfee, A.: Supplementary data for Varroa destructor: A complex parasite, crippling honeybees worldwide (2021)
20. Yeap, K.H., Koh, J.H., Loh, S.H., Dakulagi, V.: Development of a Humanoid Robot's Head with Facial Detection and Recognition Features Using Artificial Intelligence, Safety, Security, and Reliability of Robotic Systems: Algorithms, Applications, and Technologies, CRC Press, 30 December 2020, pp. 21–50. ISBN: 9781000328059 (2020)

21. Yeap, K.H., Sayago, J.: Integrated Circuits/Microchips, InTechOpen, UK, 16 September 2020. ISBN: 978–1–83968–461–6 (2020)
22. Yeap, K.H., Thee, K.W., Lai, K.C., Nisar, H., Krishnan, K.C.: VLSI circuit optimization for 8051 MCU. Int. J. Technol. **9**(1), 142–149 (2018)
23. Ziegelmann, B., Lindenmayer, A., Steidle, J., Rosenkranz, P.: The mating behavior of Varroa destructor is triggered by a female sex pheromone. Apidologie **44**(3), 314–323 (2013)

Walk Through Event Stream Processing Architecture, Use Cases and Frameworks Survey

Passent el-Kafrawy and Mohamed Bennawy

Abstract Nowadays events stream processing is one of the top demanding field(s) because of the business urgent need for ongoing real time analytics & decisions. Most business domains avail huge amount of data aiming to make use of each data point efficiently. Corporate(s) have a cloud of events vary from internal business transactions, social media feeds, IoT devices logs, ... etc. In this paper we would discuss state of the art event stream processing technologies using cloud of events by summarizing event stream processing definition, data flow architectures, common use cases, frameworks and architecture best practice. A final comparison is given for best practice and technology use based on data type.

1 Introduction

Handling a stream of *real time events* continuously, is a huge benefit for any company. Currently in most corporates, advanced analytics for business intelligence analysis is done in offline batch architecture or semi-batch architecture. This cause these advanced analytics to be delayed one day than actuals. Consequently, important insights are not on time when needed. Event stream processing helps in getting fresher insights and faster reactions. Thus decision makers will be able to give on time feedback and have full updated knowledge about the business performance and downsides.

First of all, the main component of our research is an ***event***, thus the definition of event objects (also known as "event messages", or "event tuples") are objects that are acquired as an event through a specific time window. The data included in such objects commonly include the type of the event, time of the activity, the event

P. el-Kafrawy (✉) · M. Bennawy
School of Information Technology and Computer Science, Center of Informatics Science, Nile University, Giza 12588, Egypt
e-mail: pelkafrawy@nu.edu.eg

M. Bennawy
e-mail: m.zohair@nu.edu.eg

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_3

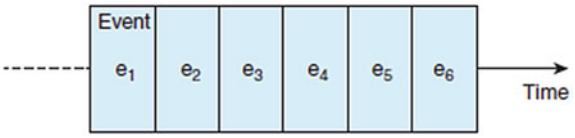


Fig. 1 A stream of events in sequence from e1 to e_n [1]

location, trigger of the event, and other metadata. Therefore, a stream consists of a series of event objects, in chronological order of arrival, Fig. 1 [1].

Events Within any Large Company can be Categorized into Three Main Types

- The first is ***business transactions***, are business related transactions or logs in different platforms. Identified by customer account describing different. KPIs.

  Examples: customer orders, insurance claims, bank deposits or withdrawals, customer address changes, telecommunication calls, delivery details, airline reservations, or transaction statements [2].
- The second is ***information reports***, external sources of data to the company that are essential for decision makers [2].

  Examples: tweets, news articles, market data, weather reports, and social media posts, Facebook and Linkedin messages.
- The third, and fastest growing is ***Internet of Things (IoT)*** data, that is data gathered from *sensors* produced by physical devices [2].

  Examples: GPS-location data from vehicles or cellular, temperature or weather data, RFID tag readings and patient monitors like CGA, motion and body temperature [2].

The purpose of performing event stream processing in any of the three types is to enable businesses to react and take decisions in real-time. The stream of events requires advanced analytics that cannot be accomplished with normal machine learning models or common enterprise support systems. Stream analytics is known as event stream processing or complex event processing.

2 Data Architecture

The history of continuous event streams is reviewed in this section providing the evolution of business data flow frameworks. The advancement of this type of. Data processing mechanisms had been evolved in three phases:

- *The classic era*—since 2003 by Ralph Kimball, data in warehouses handled in batch loads and by time became big data especially with the evolution of SaaS operating systems. In this era most of the business decision(s)/analytics were processed offline and delayed from operation, Fig. 2 [3].

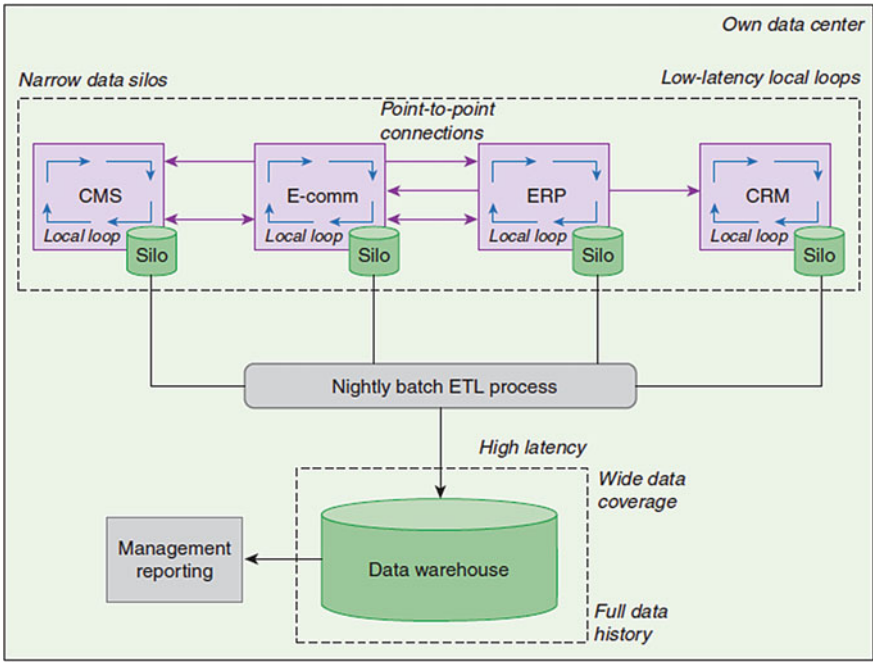


Fig. 2 Classic era diagram [3]

Data were extracted in overnight batches through ETL processes extracting data from the data silos. Then the data are transformed or modelled for reporting, to be sent back to the data warehouse. The major drawback of the classical era is the high latency in reporting decisions. The batches were usually developed based on the star schema of fact and dimension data marts [3].

- *The hybrid era*—current implementation of data flow is handled between batch loading and online streaming Fig. 3.

  In the hybrid era external SaaS dependencies were added; introduced Hadoop as a new high-latency and new low-latency data pipelines for use cases such as systems monitoring and product recommendations. Hybrid era pain points are no single point of truth, decision and solutions are fragmented and analysis either have low latency or wide data coverage, but not both [3].

- *The unified era*—evolution in the architecture by emerging online streams in a unified log and enable continuous event processing for Fig. 4 [3].

  The architecture emphasis on a unified log and a longer-term archive of events in Hadoop. The data architecture is now much simpler, with far fewer point-to-point connections, and all of our analytics and decision-making systems now working off a single version of the truth [3].

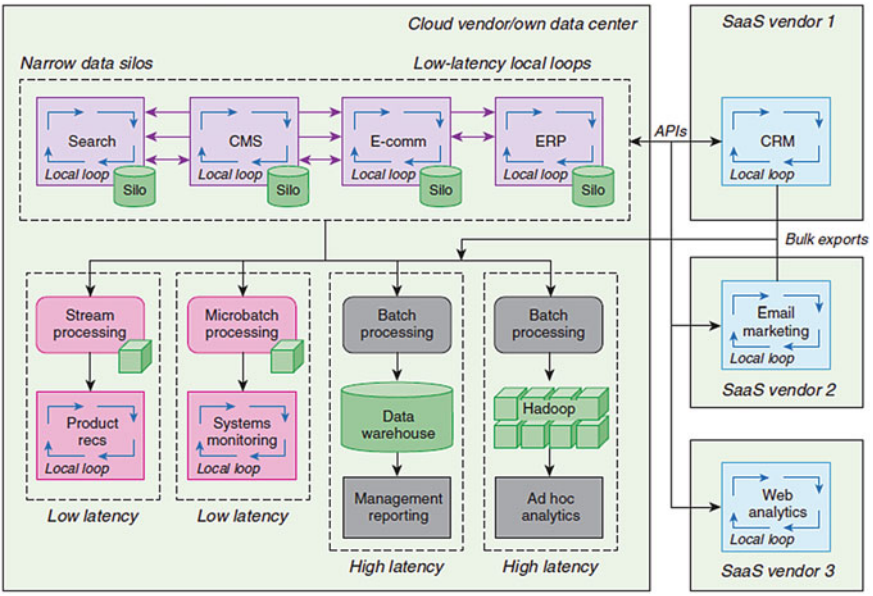


Fig. 3 Hybrid era diagram [3]

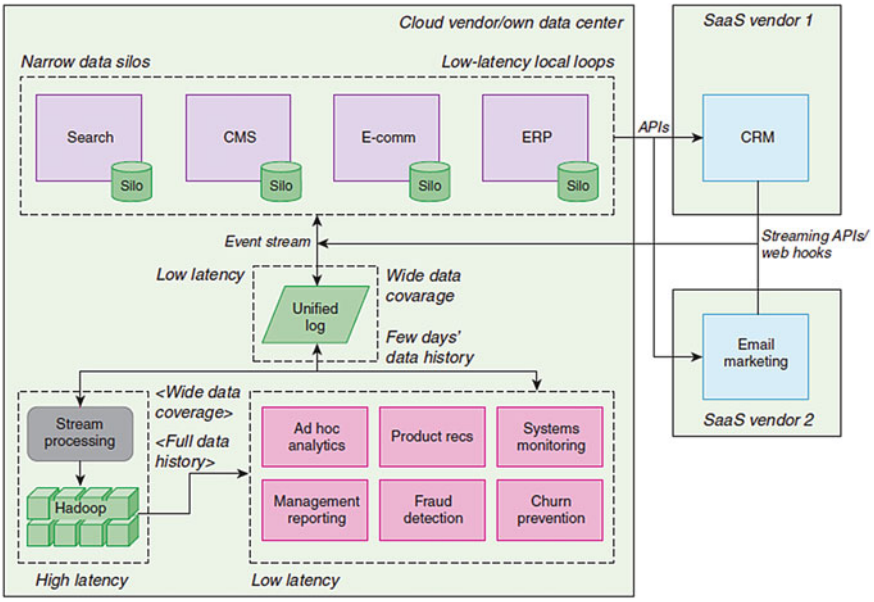


Fig. 4 Unified era diagram [3]

3 Use Cases

Making use of unified era architecture and the newly introduced streaming capabilities, a lot of use cases for event-driven applications will shine. Listed below list of use cases that will evolve with event stream processing capabilities [4]:

- Real-time recommendations, Capture events in real time and process machine learning recommendation algorithms on top of events considering event status changes (e.g., Online retail recommendation based on preferences and search contents, Songs and movies recommendation based on browsing choices, … etc.) [5].
- Detect suspected patterns in real time streams using pattern detection algorithms and complex event processing techniques (e.g., fraud detection in banks) [5].
- Online monitoring for any abnormal behavior or pattern by applying anomaly detection algorithms [5].
- Device Monitoring: A small startup rolled out a cloud-based Internet of Things (IoT) device monitor able to collect, process, and store data from up to 10 million devices. Multiple stream processors were deployed to power different parts of the application, from real-time dashboard updates using in-memory stores, to continuous data aggregates, like unique counts and minimum/maximum measurements [4].
- Fault Detection: A large hardware manufacturer applies a complex stream-processing pipeline to receive device metrics. Using time-series analysis, potential failures are detected and corrective measures are automatically sent back to the device. Early diagnoses will help reduce maintenance cost [4].
- Fleet Management: A fleet management company installed devices able to report real-time data from the managed vehicles, such as location, motor parameters, and fuel levels, allowing it to enforce rules like geographical limits and analyze driver behavior regarding speed limits. Handling congested areas [4].
- Media Recommendations: A national media company deployed a streaming pipeline to ingest new videos, such as news reports, into its recommendation system, making the videos available to its users' personalized suggestions almost as soon as they are ingested into the company's media repository. The company's previous system would take hours to do the same [4].
- Faster Loans: A bank active in loan services was able to reduce loan approval from hours to seconds by combining several data streams into a streaming application [4].
- Medical Tracking: medical providers will be able to track heartbeat, temperature, oxygen level or any IoT medical wearable devices to allow medical providers help chronic patients immediately.

4 Frameworks

Based on event stream processing methodology in handling the stream of events, ESP frameworks implement any Streaming framework in two main categories:

- ***Native Streaming***: which reflect continuous processing for the events from different streams as soon as it arrived without any latency or waiting time window Fig. 5. The continuous running processes are called operators or tasks or bolts based on the framework used. Examples: Storm, Flink, Kafka Streams, Samza.
- ***Micro-batching***: rather than processing the events on arrival , mircro-batching group events into a predefined time window called batches. Batches processed upon batch time window completion. It is known as fast batching as it is delayed few seconds than the event time Fig. 6. One of the known platforms in micro-batching Spark Streaming.

  In order to understand the main differences between the two methodologies, we need to keep an eye on fault tolerance, throughput, latency and state management.

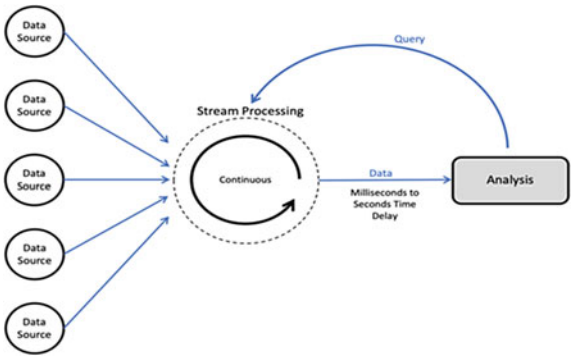


Fig. 5 Native streaming diagram

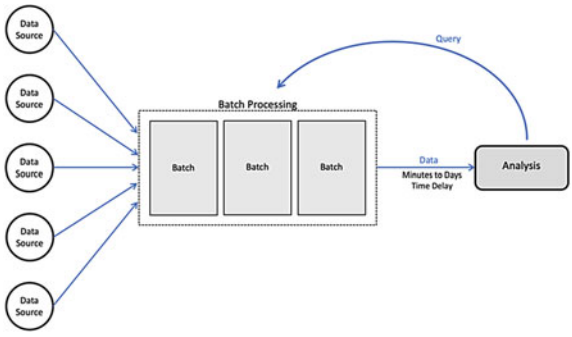


Fig. 6 Micro batching streaming diagram

In native streaming the events are processed as soon as it arrives which means minimum latency, high throughput, easy state management and hard to achieve fault tolerance.

On the other hand, micro batching methodology is quite opposite because it groups the events into a small batch and then start processing. Therefore, micro batch have higher latency, lower throughput, harder to maintain state management, and better fault tolerance.

Based on the use case business, processing logic, latency constraints and deployment environment; you would choose the best methodology and framework to work with.

Walkthrough Most Used Open Source Frameworks and a Comparison Between Each will be Discussed.

- Apache Storm

Considered as one of the state-of-the-art frameworks a decade ago. Storm address the problem of processing a huge amount of real time incoming data using a cluster of nodes. Storm is a distributed stream processing framework developed by Nathan Marz and a team at BackType using Clojure and Java. First release was on September, 2011 then project acquired by Twitter and finally released as an open-source system [6, 7].

Architecture of a storm cluster is similar to Hadoop cluster where storm run topologies similar to Hadoop MapReduce jobs except that topology process the messages forever and MapReduce process job until it finishes. Strom cluster have two types of nodes master node and worker node. Master node execute a daemon Nimbus similar to JobTracker in Hadoop, Nimbus main responsibility is to assign tasks to other nodes and monitor failures. Worker node execute a daemon called supervisor which listen to tasks assigned to the nodes and start and stop based on what Nimbus has assigned to it. In order to monitor the state and heath of the cluster a Zookeeper cluster is added Fig. 7 [7, 8].

Strom Application is considered as a topology similar to the concept of directed acyclic graph (DAG) where you have spouts and bolts as the graph vertices. Developers define a series of spouts associated with data source to load data into tuples and bolts to process the incoming data and transform it into new data stream. Both, the topology reflects the data transformation pipeline Fig. 8 [6–8]

Based on storm architecture, cluster main advantage is stability, speed, flexibility and fault tolerance. Therefore, storm can process millions of events in just seconds and is being utilized in multiple use cases ex). real time (advanced analytics, machine learning), ETL jobs, real time monitoring [6].

- Apache Heron

Apache heron is the direct successor for Apache storm by revamping storm and adding new key features. BackType team developed Heron architecture following Apache storm and the framework developed using Java and Scala. Heron framework was also acquired by Twitter and later then it is released to be an open source in 2016 [10].

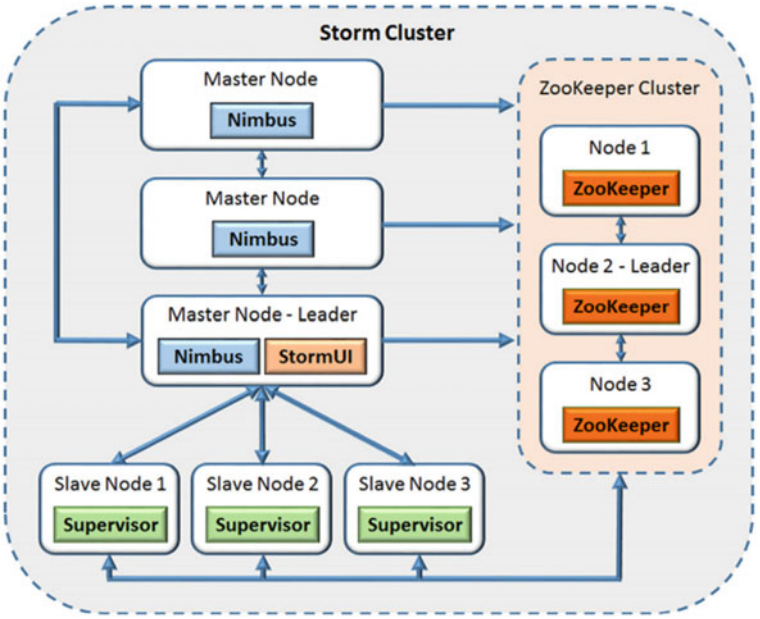


Fig. 7 Apache storm cluster architecture [9]

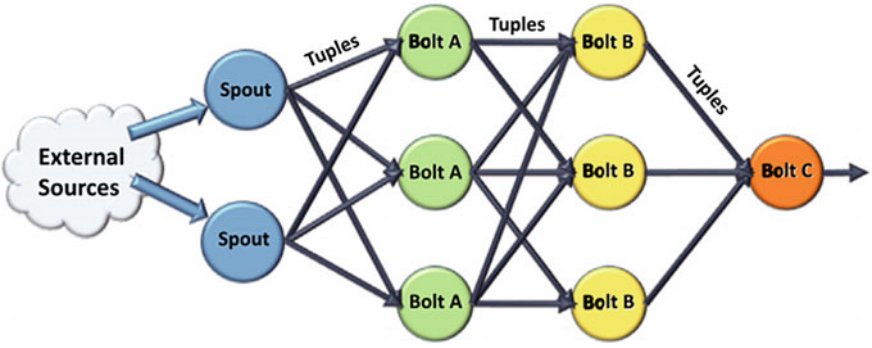


Fig. 8 Apache storm topology [9]

Heron application(s) is also designed to have two main components sprouts and bolts to connect data sources, stream data, and process each stream and finally generate output stream [6].

Talking about the key features that Heron provided over Spark.

- New Schedular: abstracting out the scheduling feature, it become easier to deploy on a data center or shared infrastructure which may use various scheduling frameworks like YARN, Mesos, or a custom environment [6, 11].
- Back Pressure Mechanism: In order to handle traffic spikes and pipelines congestions, back pressure mechanism dynamically adjust the rate of data flow in a stream during execution without affecting data accuracy [6, 11].
- Easy debugging**:** better understanding for each task profile, behavior and performance. Furthermore, quicker and better troubleshooting for issues using UI of Heron topologies [12].
- Scalability and latency: Heron stretched the processing ability to process huge number of topologies with high throughput and low latency. Concurring to a blog post by Karthik Ramasamy, "Twitter's production Heron system delivered throughput that's 10–14× higher than what its production Storm system could process" [6, 12].

Despite the fact that Heron has extra key features over storm, still storm is being used in use cases. Also, Heron is being developed to handle extra features and big data ecosystem compatibilities [6].

- Apache Samza

Apache Samza, an open-source stream processing framework. It was developed using Java and Scala at LinkedIn, then become part of the Apache Software Foundation in 2013, afterwards in 2015 Samza became a top-level Apache project [6, 13].

Samza designed to give more attention to scalability and operational robustness [13]. Samza architecture consist of main three layers a Kafka streaming layer, Yarn execution layer and Samza API processing layer, which follows the Hadoop architecture (Yarn as execution, HDFS for storage and MapReduce for processing). It is also important to highlighted that Samza architecture isn't limited to Kafka and Yarn both are pluggable layers and can be changed [14].

Samza job contains a group of tasks which is a set of Java Virtual Machine (JVM) instances, and each instance a part of the input data. Each task is a single thread process that execute over a sequence of data records. The data inputs to a job are

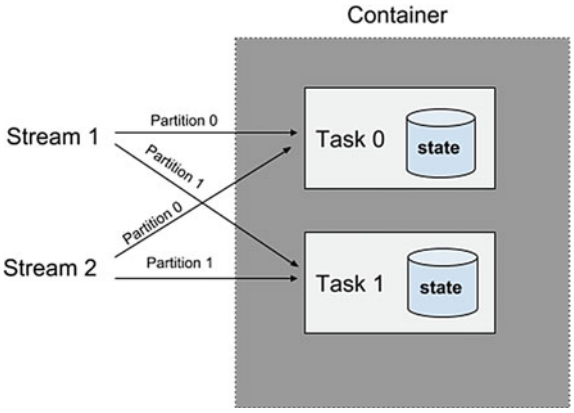


Fig. 9 Apache samza architecture [14]

portioned into disjoint subsets, and each input partition is assigned to one exactly processing task which similar to MapReduce Fig. 9 [6, 13, 14].

Based on the Apache Samza project website "A stream in Samza is a partitioned, ordered-per-partition, repayable, multi-subscriber, lossless sequence of messages," the group says [6].

If we compared the architecture of Samza and Storm, a lot of similarities will be detected. For example, Samza Streams is equivalent to sprouts in storm, tasks are similar to bolts and messages in Samza are tuples in Storm [6].

Recently in August 2020, Apache announced the release of Samza 1.5.1, which introduces new features and bug fixes [6, 15].

- Apache Spark

Apache Spark is a fast, reliable, and fault-tolerant distributed computing framework for large-scale data processing [4]. Spark consists of a core engine, a set of abstractions built on top of it (represented as horizontal layers), and libraries that use those abstractions to address a particular area (vertical boxes) Fig. 10 highlights spark main components with the areas that are used in event stream processing highlighted in blue.

o **Spark Streaming** apply Spark's distributed computing capabilities to stream processing by transforming continuous streams of data into discrete data collections on which Spark could operate. Spark introduces a new abstraction, the *Discretized Stream* or *DStream*, which exposes a programming model to operate on the underlying data in the stream [4, 6].

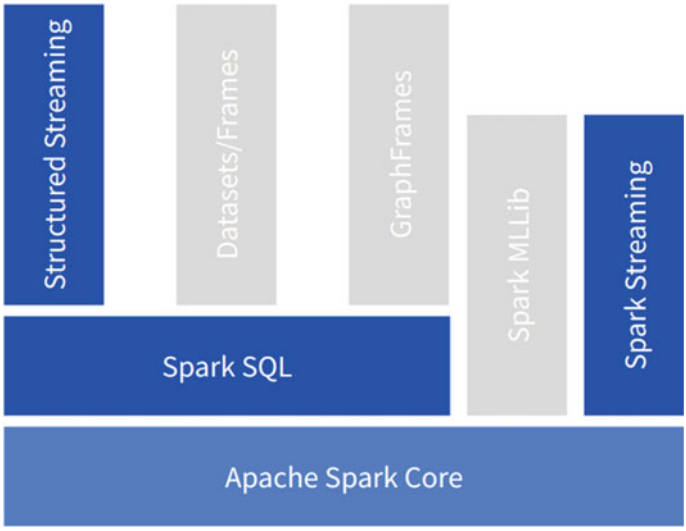


Fig. 10 Apache strom main components [4]

- **Spark Structured Streaming** is a stream processor built on top of the Spark SQL abstraction. It extends the Dataset and DataFrame APIs with streaming capabilities and inherits all the optimizations implemented in Spark SQL. It offers an API based on DataFrames and the notion of continuous queries over an unbounded table that is constantly updated with fresh records from the stream [4, 6].

Spark are available for Java, R, Python and Scala and consider are the most heavily used framework. Spark released back to 2014and the commercial venture behind spark Databrick. Recently in march 2021 Spark released version 3.1 which adds as part of Project Zen python Python dependency management support and type annotations. Other main updates improvements in ANSI SQL compliance support, history server support in structured streaming, the general availability (GA) of Kubernetes and node decommissioning in Kubernetes and Standalone [4, 6, 11].

- Apache Flink

Flink is a distributed system for stateful parallel data stream processing. A Flink setup consists of multiple processes that typically run distributed across multiple machines. Apache Flink is developed in Scala and Java, and is designed to execute arbitrary dataflow use cases in a data parallel methodology [5, 6].

Flink is compatible with cluster resource managers such as Yarn, Apache Mesos and Kubernetes. Also, it can be configured to run as a stand-alone cluster. Flink solution does not include the distributed storage but it can make use of the distributed filesystems like HDFS or object stores such as S3. For leader election in highly available setups, Flink depends on Apache ZooKeeper [5, 6].

Flink setup consists of four different components that work together to execute streaming applications and run-on Java Virtual Machines (JVMs). These components are a JobManager, a ResourceManager, a TaskManager, and a Dispatcher Fig. 11.

- *JobManager* acting as master process that controls the execution of a single application
- *ResourceManager* is responsible for managing TaskManager slots, Flink's unit of processing resources
- *TaskManagers* are the worker processes of Flink. Typically, there are multiple TaskManagers running in a Flink setup. Each TaskManager provides a certain number of slots
- *Dispatcher* runs across job executions and provides a REST interface to submit applications for execution.

Flink Table API which avail SQL like functions. Flink is being used in Netflix and other large corporare and it is one of the most promising frameworks [5, 6].

Flink was first developed as a project in German university project after that released as Apache project in 2014. Fata Artisans, the commercial vendor behind Flink and recently acquired by Alibaba [5, 6].

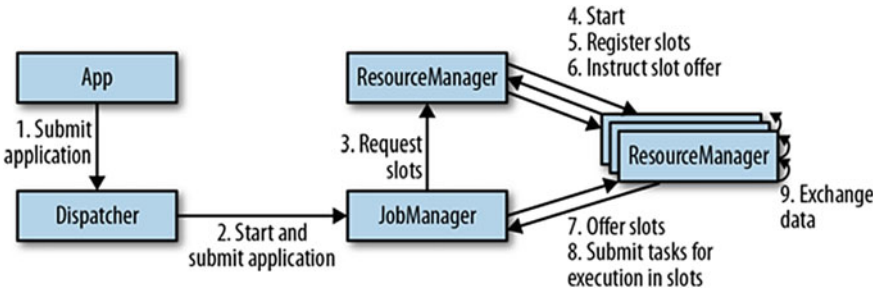


Fig. 11 Apache flink main components [5]

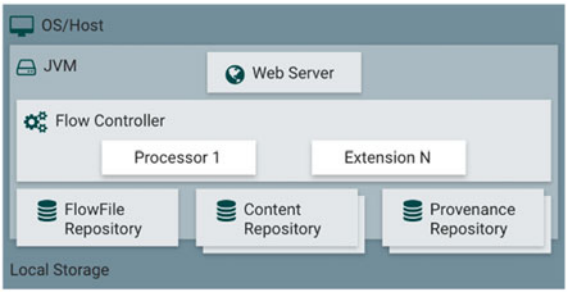


Fig. 12 Apache Nifi main components [16]

- Apache NiFi

Nifi is a Java based project, that's developed to automate the flow of data between disparate and different sources.

The product is based on the NiagaraFiles software previously developed by the US National Security Agency (NSA), and was released in 2014 as an open-source project [6, 16].

Nifi has six main components, *Web Server*, *Flow controller* (Controls the running of Nifi extensions and schedules allocation of resources), *Extensions* (plugins to interact with different systems), *FlowFile* Repository (track status of current active Flow files), *Content Repository* (place in local storage where the content of all FlowFiles exists and it is typically the largest of the three Repositories) and *Provenance repository* (history of each flow file is stored) Fig. 12 [6, 16].

NiFi developed in java with the ability to operate within a cluster, main advantages of Nifi over other platform security, usability (portal to visually monitor and modify behaviors. NiFi is backed by Cloudera, which acquired Hortonworks, which in turn acquired Onyara, the company that developed NiFi. Nowadays the software is developed via the Apache NiFi community [6, 16].

5 Comparison

Illustrate the main features in each event stream processing framework with advantages and disadvantages [4, 5, 17–21].

As all frameworks are open source, the comparison shows the release date for each framework, Owner (initial development team) and commercial sponsor. Another KPI is the development programming language (used to develop framework) and

supported programming languages (developers will use it while using framework). SQL Support feature is an API that translate the data streams into a logical layer and gave developers ability to type SQL like queries to explore the stream. Finally the supported time windows, key features, disadvantages and deployment mechanism for each platform.

	Storm	Heron	Samza	Flink	Nifi	Spark Streaming
Released Open Source	2014	2016	2014	2014	2014	2014
Developed Programming Language	Clojure and Java	Java and Scala	Java and Scala	Java and Scala	Java-based	Scala
Owner	BackType / Twitter	BackType / Twitter	LinkedIn	German university project, commercially data Artisans acquired by Alibaba. Used in Netflix	National Security Agency, backed by Cloudera	University of California, commercial Databricks
Supported Programming Languages	Java	Java	Java	Java , Scala , Python	GUI not APIs	R, Python, Scala, and Java
SQL Support	Storm SQL		Samza SQL	Flink SQL	N/A	Spark SQL
Window	Fixed , Sliding & Global	Fixed , Sliding & Global	Fixed , Sliding , Session & Global	Fixed , Sliding , Session & Global		Fixed & Sliding
Key Features	- fault tolerant	- scheduler	reliable persistency with low latency	-architected probably	- Ease of Use	- heavily used
	- flexibility	- scalability		-sophisticated state management	- Flexible Scaling Mode	- data frame API
		- latency		- Low latency & High throughput	- Security	- Fast recovery
						- Better load balancing
Disadvandges	-does not guarantee state consistency	lack of support	Only supports JVM languages	scaling limitations	-state persistence issue	-micro-batching affect latency
	- low throughput		Does not support very low latency			-no session windows
Deployment	Apache Zookeeper		Standalone , Hadoop Yarn	Standalone , Hadoop Yarn , AWS , MapR , Docker ,Kurbernetes		Standalone , Hadoop Yarn , Apache Mesos , Kurbernetes

6 Conclusion

Event stream processing will be part of any data architecture as most corporates are moving from the classic era to the unified era. Corporates would like to make best of the real time decisions and insights rather than offline batches in the current classical era. Therefore, we introduced top use cases and frameworks that would benefit from the real time decision(s) produced from event stream processing capabilities. Regarding frameworks selection, multiple factors including deployment, ecosystem, security… etc. would decide the perfect framework for your application and a detailed comparison is shared to illustrate the main differences between the listed frameworks. Finally, one of the very promising use cases is usage of event stream processing in real time recommendations which will be studied further in the future.

References

1. https://complexevents.com/2020/06/15/whats-the-difference-between-esp-and-cep-2/
2. https://complexevents.com/when-do-you-need-an-event-stream-processing-platform/
3. Event streams in action real-time event systems with kafka and kinesis by alexander dean, Valentin Crettaz (z-lib.org). ISBN: 9781617292347
4. Stream Processing with Apache Spark. ISBN: 978–1–491–94424–0
5. Stream Processing with Apache Flink. ISBN: 978–1–491–97429–2
6. https://www.datanami.com/2019/05/30/understanding-your-options-for-stream-processing-frameworks/
7. https://en.wikipedia.org/wiki/Apache_Storm
8. https://storm.apache.org/releases/current/Tutorial.html
9. https://www.researchgate.net/publication/319680334_Aging-related_Performance_Anomalies_in_the_Apache_Storm_Stream_Processing_System
10. https://heron.incubator.apache.org/
11. https://spark.apache.org/releases/spark-release-3-1-1.html
12. https://blog.twitter.com/engineering/en_us/a/2015/flying-faster-with-twitter-heron
13. https://martin.kleppmann.com/papers/samza-encyclopedia.pdf
14. https://samza.incubator.apache.org/learn/documentation/latest/introduction/architecture.html
15. https://engineering.linkedin.com/blog/2018/11/samza-1-0--stream-processing-at-massive-scale
16. https://nifi.apache.org/
17. Making sense of stream processing-the philosophy behind apache kafka and scalable stream data platforms, Martin Kleppmann. ISBN 978–1–491–93728–0
18. https://www.upsolver.com/blog/popular-stream-processing-frameworks-compared
19. https://flink.apache.org/news/2016/04/06/cep-monitoring.html
20. https://complexevents.com/2019/06/09/eight-trends-in-event-stream-processing-may-2019/
21. https://www.upsolver.com/blog/streaming-data-architecture-key-components

Artificial Intelligence for Cybersecurity: Recent Advancements, Challenges and Opportunities

Veenu Rani, Munish Kumar, Ajay Mittal, and Krishan Kumar

Abstract With the advancement of technology, there exists a wide variety of cyber-security attacks like ID theft, cracking the captcha, data breaches. These attacks are affecting individuals as well as well-established organizations. To handle these attacks, a robust and intelligent system is required. Artificial intelligence (AI) is one of the most emerging areas used in cyber security to protect internet-connected systems from eavesdropping, attacks, unauthorized access, and threats. AI is a technique that enables machines to tackle every situation intelligently. AI has been used almost in every area, from health care to robotics. The concept of AI in cyber-security makes machines more intelligent and actionable compared to traditional approaches. In this article, we focus and deliberate concisely on the use of AI in cyber-security, its application, various challenges, and opportunities. Authors have also highlighted future perspectives of AI in cybersecurity.

Keywords Phishing · DOS · Cyber-security · Cyber-cell

1 Introduction

Intelligence is the only term that makes humans different from anything else in this universe [7]. Instead of human's natural intelligence, researchers started working on "can a machine think like human beings?" This concept attracted many researchers around the world, and the result is "Artificial Intelligence." The great researcher Alan Turing is known as the father of AI, who has performed a Turing test to measure the

V. Rani · M. Kumar
Department of Computational Sciences, Maharaja Ranjit Singh Punjab Technical University, Bathinda, Punjab, India

A. Mittal (✉) · K. Kumar
University Institute of Engineering and Technology, Panjab University, Chandigarh, India
e-mail: ajaymittal@pu.ac.in

K. Kumar
e-mail: k.saluja@pu.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_4

intelligence level. AI is a fast emerging branch of computer science that deals with making computers intelligent. It is an issue of conflict between researchers,some argue that AI can behave and reason like humans, and others oppose it [35]. AI has a vast application area in almost every domain like face recognition, speech recognition, robotics, and many more [23]. Except in these areas, AI has performed excellently in the field of Cybersecurity. As in the present era, dependency on digitization and the internet has increased; various security threats such as unauthorized access, denial of service, phishing, spoofing, and others have also increased from last years [32]. These types of attacks can cause financial loss for an individual as well as an organization. According to a report by Norton, the global cost of data breach recovery is $ 3.86 million, and companies require 196 days to recover from a data breach attack. The increasing rate of cybercrime has given an alarming message to cybersecurity professionals [33]. Cybersecurity is a branch which deals with the above-said attacks and promises to provide security to the government, businesses, organizations, and individuals. Cyber-security systems comprise network and computer security systems [9]. For the storage of a significant amount of data on computers, intelligent cyber security is essential nowadays. In the past, antivirus, firewalls, and encryption methods were used to protect our systems, but these are not effective these days [6]. The main problem with these conventional methods is that a few security experts operated these methods. During the Covid-19 Pandemic, every operation has been performed online. This digitization has increased the risk of cyber-attacks. In this new environment, cybersecurity professionals applied more AI-based techniques to reduce these attacks. The word cyber is taken from the Greek word Kybereo, which means to steer, guide, control, and was first used at the end of 1940 by Norbert Wiener. He used the Cybernetic word to describe computerized control over machines and living organisms [42]. In 1949 John Von Neumann first speculated that a computer program could be replicated, and a computer virus was made in the form of a program. At that time, only computer viruses were considered cyber-attacks. In 1950 phone phreaking came into existence. In the mid of 1960, the first malicious hacking was introduced in the institute of Massachusetts. At that time, computers were huge in size, and students hacked the common system without any commercial or geopolitical benefits. In the real sense, Cybersecurity began in 1972 with the advent of ARPANET, when Bob Thomas invented the first network virus named Creeper. With the birth of Email in 1990, cybercrimes are spreading very fast across the world.

Now Cybersecurity has entered into the fifth generation [31]. Many organizations have adopted AI-based applications to handle these severe issues, intelligently protecting cyberspace [43]. AI-based techniques can excellently monitor and combat cyber-crimes and threats. Artificial Intelligence performs three main functions in Cybersecurity: Detection, Prediction, and response. Over 50% of organizations are implementing AI for the detection of cyber-security attacks. Deep learning techniques are used to detect the irregular behavior of official authorities. 35% of organizations are making extensive use of AI to predict cyber-attacks. AI-based systems automatically sense the network topology, identify critical vulnerabilities and predict any

malware attack. When responding to threats, only 18% of organizations implement AI to react to attacks.

1.1 Motivation

The main motivation is to represent a brief discussion on the concept of AI and cyber security. During the Covid-19 pandemic, the increased use of the internet has also increased the chances of cyber-attacks. In this article, we have discussed the meaning of cyber-security and how AI can resolve the various types of attacks. As we have reviewed many papers related to this topic, our article provides a detailed description of the cyber-security term and how it has come into its origin. Different types of challenges faced by researchers are also discussed in this article. As per the Scopus database, we discussed year-wise publications which were not covered in the past. AI is a technique that makes machines intelligent to decide without human intervention. When AI incorporates with cyber security, it can handle unpredicted attacks, making it a broader area for researchers.

2 Related Work

AI in cyber-security is a novel field, as there is not enough literature on this topic. From 2010 to 2021, only 297 articles were found on the Scopus database with the searching keyword "Use of AI in Cyber-security." Year Wise total number of published articles from 2010 to 2021 are shown in Fig. 1. Some latest work done by researchers in this field is described here. Sedik et al. [34] have used deep learning Convolution Neural Network (CNN) model to implement cyber-security in 5G Networks

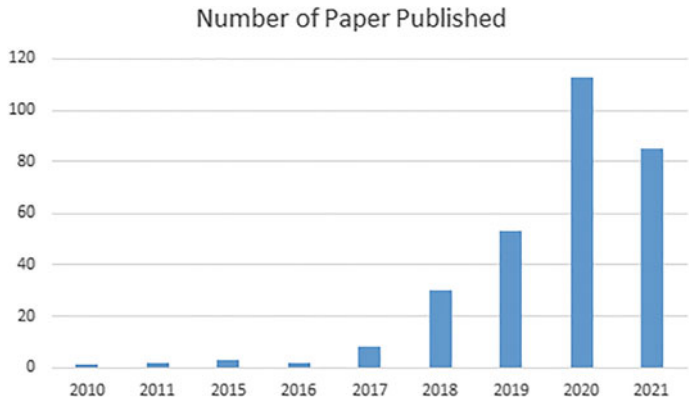


Fig. 1 Year wise literature published

based secure Smart cities. Their proposed method mainly focus on various security mechanism used in human identification. In their article they have tried to discriminate between original and fake biometrics. Latif et al. [15] have proposed a protocol to create a chain block chain framework to secure data transmission among IoT devices. They have neglected classical cryptography techniques like hash function, quantum hash function for linking blocks of the chain. Gupta et al. [17] have applied machine learning and smart card based factors for authentication purpose. Their proposed model checks various types of attacks and also diagnose their functionality. Authors have developed a new approach named Telecare Medical Information System (TMIS) to enhance secure communication between doctors and their patients. Zhang et al. [48] have presented a systematic review on Artificial Intelligence in cyber security. How AI is spreading its wings in the cyber security domain, what challenges researchers face, everything has been explained in their article. Authors have also noticed the limitations of AI in the cyber security domain. Geetha and Thilagam [16] have written a review paper on the effectiveness of Machine Learning (ML) and Deep Learning (DL) algorithms for cyber-security. They have presented a detailed classification of several algorithms of ML/DL in their article. They have also described numerous cyber-attacks like phishing, hacking, malware, denial of service (DOS), SPAM, etc. Tao et al. [38] have discussed the future of Artificial Intelligence in cyber security in their article. They have defined quantitative and qualitative approaches to map different areas of Internet of Things (IoT), cybercrimes, business, Intrusion Detection System (IDS), and cyber forensics. In their article, they have found that computer complexity, and threats of cyber-crimes can be reduced by incorporating AI in this domain. Naik et al. [27] have described various applications of AI in cyber security. In their article, the authors have presented how AI can analyze, detect and take action against cyber-attacks. They have also discussed the future scope of AI and the various challenges researchers face. Salhi et al. [29] have demonstrated a study on E-mail classification for forensics analysis. They have described that most of the networks attacks are affected by e-mail these days. In their article, authors have presented a new method for e-mail clustering to segregate them into good and bad ones. Xu et al. [44] have used Federated Learning (FL) in cyber security. They have proposed a multi agent FL to allocate security policies among multiple devices without the requirement of sharing policy information. Radanliev et al. [28] described the implementation of AI in the cybersecurity domain. In their article, they have identified a dynamic approach in which AI and machine learning techniques are used for cyber risk analysis. Their approach is self-adapting in nature that predicts cyber risks and threats when new edge computing nodes are deployed. Chaudhary et al. [10] have presented a review of various challenges in cyber-security using AI. They have called AI a double-edged sword that can improve security across the network and create new forms of attacks. They have mentioned various types of attacks as well as their detection system using Machine learning algorithms. Authors have shown that the Support Vector Machine (SVM) model can provide a 99% accuracy rate compared to other models. Ali et al. [4] have discussed various ML classifiers used in cybersecurity like Random Forest (RF), Decision Tree (DT), Support Vector Machine, and Artificial Neural Network (ANN)

in their article. They have done meta-data analysis on existing literature of AI applications in cybersecurity. Salloum et al. [30] have explained various machine learning and deep learning techniques for network analysis of intrusion detection. They have also described numerous datasets like KDD Cup 1999, ISOT dataset, HTTP CSIC 2010 dataset, CTU dataset, and UNSW-NB 15 dataset used in artificial intelligence for cyber-security. Eian et al. [14] have described cyber-attacks during Covid-19 and their possible solutions. Authors have stated that online activities have been increased by nearly five times during pandemics, which are more vulnerable to cyber-attacks. They have presented reports of various organizations (WHO) depicting how cyber-attacks affected individuals and IT companies during the Covid-19 Era. Authors have discussed that AI is the unique solution to tackle these attacks during such an emergency time. Shidaganti et al. [36] have presented a model for the prevention of Distributed denial of service (DDoS) attacks from the cloud. In their article, the authors have mentioned that DDoS attacks are the biggest threats to the network. With the advent of cloud computing, these attacks have become harder to trace. Authors have proposed a selective cloud egress filter (SCEF) model, which implements specific modules to deal with these types of attacks. Ubale and Jain [40] have presented a survey on DDoS attacks. They have discussed various techniques of DDoS attacks and their solutions and Software-defined networking (SDN) and its advantages over traditional networks in their research work. Malatji et al. [24] have discussed the impact of AI on human life and cyber security. They have discussed the two phases (positive and negative) of AI. In their article, authors have presented arguments in favor of an AI-based system that does not require human interference and the ideas that counter the above-said words. Brundage et al. [8] presented that AI-based systems are more susceptible to numerous security vulnerabilities different from traditional methods. Li [22] has discussed the positive and negative face of AI in cyber security. He has reviewed that by using deep learning techniques of AI, threats of cyber-attacks can be controlled. On the other hand, he has also mentioned that the advent of AI has increased the number of cyber-attacks.

3 Framework of AI Based Cyber-Security System

This section will describe the framework of AI-based cyber-security system, which is the basis for understanding how AI may be implemented in this field. As AI works on the concept of perception, it means using previous knowledge and sensing the environment decision can be taken. So the first requirement in a data science-based system is a database. After collecting data, machine learning techniques are applied to quantify cyber risks, analyze their behavior, and generate alerts regarding risks. This all process is shown in Fig. 2.

The general step is to collect data from diverse sources according to specific domains that may be classified into different categories like host, hybrid, and network [41]. After collecting the raw data, the data is refined by applying various processes. Only useful data is kept, others should be removed in this step. High-quality data is

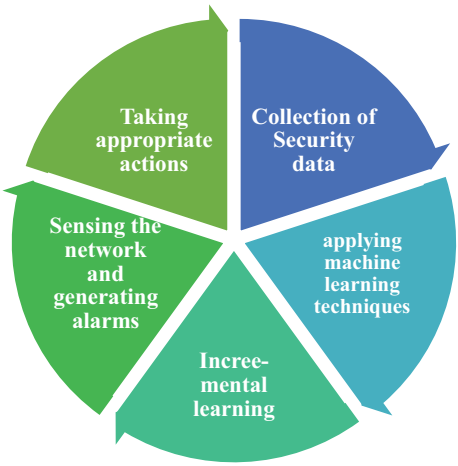


Fig. 2 Framework of AI based systems

must for achieving high accuracy. Now data is ready for applying various AI techniques to change the cyber-security landscape. Various supervised and unsupervised feature extraction techniques, features reduction techniques, classification techniques are applied to solve several cyber-security problems. Popular machine learning algorithms like ANN, Convolution Neural Network (CNN), K-nearest, Random Forest, etc., are used to sense the network. If any discrepancy is found, then intelligent machines send alarming messages to the concerned organization, and appropriate actions are taken. Each layer has a significant role in this framework. The output of one layer acts as input for another layer. Chowdhury et al. [12] have proposed a Bio-inspired hybrid framework for cyber-security composed of three AI methods. In their method, they have used hybrid ECIMSD methodology to detect malicious malware. They have collected 2598 packed viruses from the Malfease project dataset for experimental work. Authors have reported 99% accuracy in their method of malware detection.

4 Latest Trends of AI in Cyber-Security

In this section, applications of AI in cybersecurity have been described. AI is a technique that aims to make computers intelligent. It is a newly emerging area that has been used in almost every domain. By researchers, AI is defined as the science that tries to discover intelligence among the machines and makes them able to apply logic and intellectual power to make decisions, as humans do. The rapid development of the internet has changed our lives and way of working.

In recent years, AI has revolutionized many fields such as natural language processing, image processing, computerized robotics, expert systems, and many others. As every coin have two folds [39]. AI also has some pros and cons in each

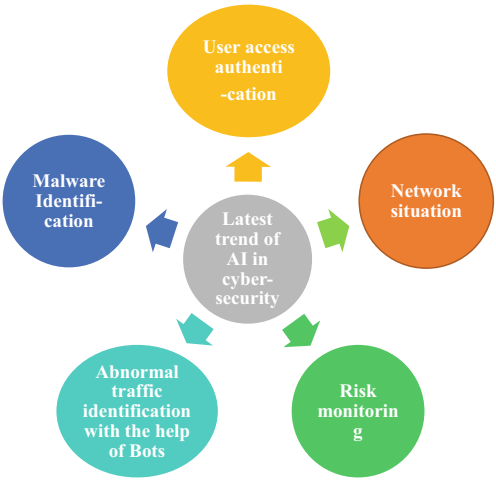

User access authenti -cation
Malware Identifi- cation
Network situation
Latest trend of AI in cyber- security
Abnormal traffic identification with the help of Bots
Risk monitorin g

Fig. 3 Latest trends of AI in Cyber-security

area that can be understood by looking at its applications. Major trends of AI in cyber-security are shown in Fig. 3.

4.1 User Access Authentication

Before accessing the network resources, it is mandatory to verify the authenticity of users. In terms of cyber-security, the system should be robust to identify all kinds of illegal behaviours. AI-based biometrics systems using the unique physical properties of an individual enhance the password matching process. The user authentication process is shown in Fig. 4.

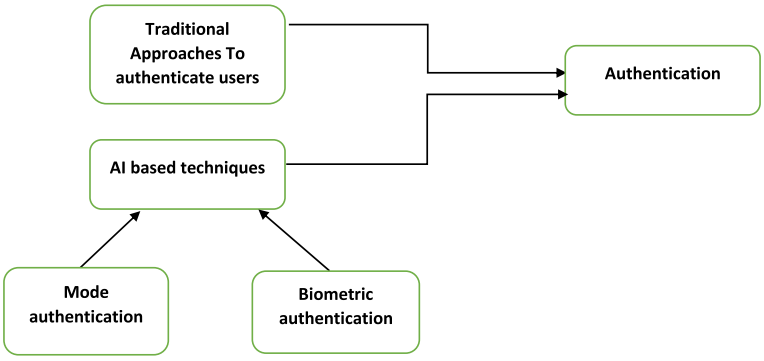


Fig. 4 User authentication process

In the above figure, user authentication is done in two ways, one is the traditional way, and another is the AI-based technique, which is further comprised of mode authentication and Biometric based. Biometric physical characteristics of individuals are used to verify their identity. Many researchers have used different physical characteristics to verify a user, like [20] fingerprint biometric to verify authenticated users. Kumar et al. [19] have used gait patterns for human identification. They have applied various machine learning techniques to recognize human gait patterns. In mode authentication, PIN codes are used for identity purposes, but that single-mode does not offer as much security [2]. Multimode authentication is used to avoid the shortcomings occurring in a single mode. Shoufan et al. [37] has used the Random forest method to achieve multimode authentication. Korkmaz [18] has a neural network to provide multimode authentication. The author has not matched only the password but also provides some variations to keyboards. These variations include the combination of the user's typing speed, typing style, and key combination to verify the user's identity.

4.2 Network Situation Awareness

The network plays an essential role in cybersecurity. The selection of weak network topology can enhance the chances of cyber-crimes. It becomes necessary to inform the network designers about weak network links, network adjustment, complex networks, and other designing issues. To achieve this process, a sound network awareness model is required with a solid knowledge base from which the condition of the network can be detected and matched. The model should be able to handle those situations which have not occurred earlier. AI-based techniques are very good in handling all these issues. Dongmei and Jinxing [13] have used Wavelet Neural Network (WNN) to aware network designers about network situations. To calculate the situation of the network, they have used a particle swarm algorithm to optimize the parameters of WNN. Their algorithm promised to reduce the data attributes in a big data environment. To assess the situation of the network [26] have used the Bayesian network, and to generate results, fuzzy risk estimation has been used.

4.3 Risk Monitoring

As new technologies like big data and cloud computing emerge day by day, the risk of hacking data is also constantly increasing. As the use of the internet and volume of data is increasing day by day, hackers have become more active and search to launch new attacks at any time. AI-based systems can be used to spot these new attacks. Traditional intrusion detection systems (IDS) cannot cope with these new attacks created every second. To handle large volumes of high-speed data and to detect hackers' activities, it has become evident that an intelligent system is

required to detect risks in advance. By using machine learning algorithms, systems can be trained to detect malicious activities. Worldwide latest knowledge regarding new cyber-attacks can be acquired through AI-based cyber-security systems that can be formulated to make decisions based on what attacks are occurring and what attacks can occur. Marir et al. [25] have pointed out a network irregular behavior detection method. They have used multilayered Support Vector machines (SVM) combined with deep feature extraction methods to reduce the dimension of large-scale distributed networks and find the irregular behavior of networks. Aljamal et al. [5] have proposed a hybrid IDS based on machine learning. In their proposed method, they have used SVM as a classifier and K-means algorithm for clustering. They have applied their method to the UNSW-NB15 dataset and achieved 84.7% accuracy.

4.4 Abnormal Traffic Identification with the Help of Bots

A network carries multiple data transformations at the same time. At normal traffic load, it can behave normally and provide high-quality services to users. However, hackers are continuously sensing the network and injecting a large amount of illegal data, making the network slow. Traffic on the network cannot be handled with manual responses alone. AI-based systems can make a difference between original and malicious data. Bots are the best example of AI-supporting systems. Ahmed et al. [3] have divided anomaly detection methods into four categories: classification, statistical, information theory, and clustering. They have used supervised and unsupervised techniques to find anomalies in the network. Zeng et al. [47] have used deep learning methods to identify end-to-end traffic. They have used a combination of CNN and LSTM to extract features from raw traffic across the network. The authors have reported a 99.98% accuracy rate in this method.

4.5 Malware Identification

Malware is a term used for various malicious attacks such as viruses, worms, Trojan horses, botnets, and others. It is a widely used method of cyber-attacks adopted by hackers. A large amount of digital data is affected by malware attacks. Researchers are adopting AI techniques to prevent and diminish malware. Xu et al. [45] have used machine learning techniques for malware detection. Authors have used SVM and random forest classifiers for the classification of the data stored in virtual memory. They have used the RIPE benchmark data set for experimental work and achieved 99% TPR. Chowdhury et al. [11] have used multilayer perceptron (MLP) and data mining techniques to detect software malware. Authors have used signature-based and anomaly-based methods to detect malware. The authors have collected 52,185 executable files for experimental results. Ye et al. [46] have used a deep learning technique to detect malware. Authors have proposed a heterogeneous deep learning

framework in which Auto Encoder (AE) combined with multilayer restricted Boltzmann machines (RBMs) has been used to discover unknown malware. For experimental work, the authors have collected files from the Comodo Cloud security center. They have shown that deep learning methods can achieve better performance than other traditional ones.

Above discussed fields are a very limited area of AI in cyber-security. However, it has vast scope in this field. AI can analyze large volumes of data with speed, accuracy, and efficiency because of its data analysis and powerful automation capabilities. AI-based systems can estimate future cyber-attacks by understanding the history of the network and its malicious attacks even though the pattern of attacks changes. Compared to conventional technology that only focuses on known attackers and attacks. On the other hand, intelligent systems track the networks. If any unprivileged activity threat is found, the machine will become more sensitive and alarm concerned organizations. During Covid 19 Pandemic period, AI has shown drastic changes in its usage. International organizations and scientists started to apply new AI technologies to track the position of viruses.

5 Challenges Faced by Researchers

While studying the implementation of AI in cybersecurity, many challenges are faced ranging from data collection to take decisions. The main challenges of AI in cyber security are shown in Fig. 5. Summary of challenges are given below:

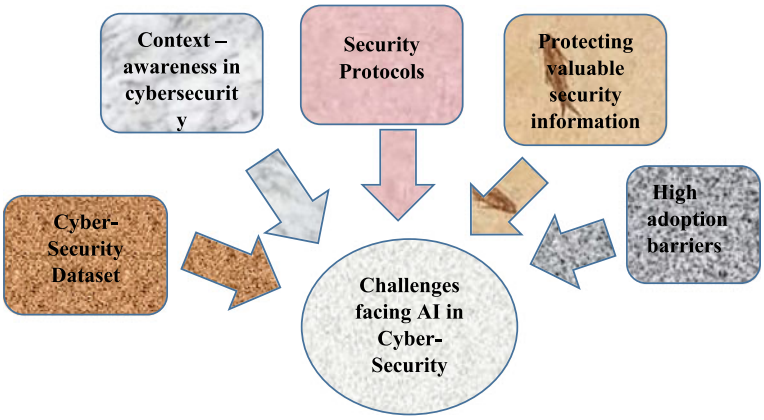


Fig. 5 Major challenges in AI

5.1 Cyber-Security Datasets

Primary component of any research is the availability of datasets. In the domain of cyber-security, most of the data is old and insufficient. It is difficult to understand the behavior of cyber-attacks from this incomplete data. Artificial Intelligence requires a proper dataset to operate. Although these datasets can be transformed into meaningful data, it requires many steps for conversion. The accuracy rate of machine learning algorithms may be low by using these types of data. Collecting real-world data is not as easy because data sources are dynamic in nature. Thus, collecting different types of data (structured, semi structured, Meta-data, etc.) related to a particular domain is challenging [33] . Therefore, collecting and managing data for effective working in cybersecurity could be a significant challenge.

5.2 Security Protocols

The protocol is a set of rules and regulations that should be followed while transferring data on a network. Hackers mostly use different programs to conceal their IP address, such as Virtual Private Network (VPN), Proxy servers, Tor browsers, etc. This type of program breaks the protocols of communication because hackers remain anonymous and undetected. Security protocols also include the common and more meticulous rules that are used to filter incoming data in sequence during the execution. Kayes et al. [21] defined cybersecurity protocols as static in nature generated by human expertise.

5.3 Protecting Valuable Security Information

Cyber-security decreases the risk of crucial data, but cyber-attacks have increased and become more dominant [1]. The major challenge of cyber-attack is the loss of valuable information while transferring it across the network. Hackers are also awaking about the latest AI-based techniques. They are developing more advanced attacks to acquire information. Like data protectors, criminals can also monitor and apply AI techniques to find why their cyber-attacks have ended in failures. Biometrics authentication is considered a blessing for protecting information from unauthorized users. However, biometric information can fall into the wrong hands and be used to infringe any organization's personal information. This has become a significant challenge for AI experts.

5.4 High Adoption Barriers

Companies require much time, money, computing power, skilled staff, and data centers to implement AI-based techniques. According to a report of IBM, (From Roadblock to Scale: The Global Sprint towards AI) skills gap is an essential barrier in adopting AI in cybersecurity. By practically security threats are growing in numbers by the passing of each day. IT industries are struggling with a security skill shortage.

5.5 Context—Awareness in Cybersecurity

Existing cybersecurity techniques work primarily for low-level features. With the advancement of new data mining and machine learning techniques, an advanced pattern is required to describe cybersecurity terms more broadly. Macqueen, professor of the University of California, has used temporal and spatial relationships to sense the network whether a suspicious activity could have occurred on the network. There is a limitation of traditional approaches because those techniques are not using contextual information for predicting attacks. Therefore lack of a context-aware adaptive system is also an issue in cybersecurity.

6 Opportunities of Implementing AI in Cyber-Security

Artificial Intelligence endeavours to generate alerts for threats, identify new types of malicious attacks and protect organizations' sensitive information. According to Dan Patterson, 2,00,000 cyber-attacks are faced by middle-sized organizations each day. These middle-sized organization cannot handle this volume of threats. AI is an ideal cyber-security solution for organizations looking to thrive online these days. AI provides many benefits of integrating AI with cyber-security.

- Human beings are not able to predict all the threats that a company may face. Hackers are launching new threats every second. These unknown threats can cause financial as well as other massive damage to a network. AI is one of the best solutions to map and take action against these threats.
- As we have mentioned earlier that, AI can perform three main functions in cyber-security. It is rightly said that detection of a disease is the prevention of it. In a similar way, detection of threats is the beginning of protecting your computer's network. AI scans our whole system and responds to unpredicted threats, which is impossible for human beings.
- Attackers change their ways of attacks very often. However, the basics of security practice remain the same. If a person tries to handle these threats, they will get bored by repeating the same process. However, AI mimics the best qualities of

human beings and can analyse the attacks in the best manner without getting bored.

These opportunities, as mentioned above, are very little compared to the dimension of AI. It provides more chances for cyber-security experts to show their skills in comparison to the traditional techniques.

7 Conclusion

In a scenario where machine intelligence and cyber-attacks are rising constantly, advance cybersecurity techniques cannot be ignored. As we have studied much literature regarding the use of AI in cybersecurity, it is evident that AI-based systems are desperately needed to understand the network situation, control the information, and various malicious attacks. As human beings cannot protect the dynamic enterprise attack surface, AI offers much-required investigation and threat identification that cyber-security professionals can use to improve security. The main reason for adopting AI in cybersecurity is the increasing complexity of threats. Before the advent of technologies, cyber threats were like a virus, Trojan horse, malware, eavesdropping etc. Simple protection methods were used to handle these types of attacks. However, hackers are intelligent these days, so new and intelligent techniques are required for handling risks. In this article, we have discussed the application area of AI in cybersecurity from the point of user access authentication, network situation awareness, risk monitoring etc. We have also covered the various challenges faced by researchers. Future perspectives of AI in cyber security are also discussed in this article.

8 Future Perspectives

Researchers have predicted that the twenty-first century is the era of Artificial Intelligence, and this prediction is becoming true. The industry is unbalanced or uneven, with the troublemakers having the chance to pick from a large number of weaknesses to dispatch their assaults, alongside sending a consistently expanding armory of devices to sidestep recognition whenever they have penetrated a framework. While they must be fruitful once, the security specialists entrusted with safeguarding a framework need to shut down each assault without fail. As the World is moving towards digitization, there is no surprise that cybersecurity has become a priority for all organizations. One has to be vigilant 24×7 to implement cybersecurity. Human experts are failing; merging AI in the cybersecurity domain has shifted the load from humans to machines that can do dogwatches on hackers' whole day without getting tired. If we say that AI will change cybersecurity, then there is nothing wrong with this. At this time, there are lots of issues that need an intelligent solution. In the future

application of AI will promise new principles to handle a large volume of data. If AI techniques are implemented correctly in cybersecurity, then that era is not so far when artificial intelligence will replace human intelligence.

References

1. Abbas, N.N., Ahmed, T., Shah, S.H.U., Omar, M., Park, H.W.: Investigating the applications of artificial intelligence in cyber security. Scientometrics **121**(2), 1189–1211 (2019). https://doi.org/10.1007/s11192-019-03222-9
2. Adekunle, Y.A., Adebayo, A.O.: Holistic exploration of gaps vis-à-vis artificial intelligence in automated teller machine and internet banking **12**(19), 4–8 (2019)
3. Ahmed, M., Naser, M.A., Hu, J.: A survey of network anomaly detection techniques. J. Netw. Comput. Appl. **60**, 19–31 (2016). https://doi.org/10.1016/j.jnca.2015.11.016
4. Ali, R., Ali, A., Aleem, S.: A systematic review of artificial intelligence and machine learning techniques for cyber security a systematic review of artificial intelligence and machine learning techniques for cyber security (2020). https://doi.org/10.1007/978-981-15-7530-3
5. Aljamal, I., Tekeoglu, A., Bekiroglu, K., Sengupta, S.: Hybrid intrusion detection system using machine learning techniques in cloud computing environments. In: Proceedings 2019 IEEE/ACIS 17th International Conference on Software Engineering Research, Management and Application, SERA 2019, pp. 84–89 (2019). https://doi.org/10.1109/SERA.2019.8886794
6. Anwar, S., Zain, J.M., Zolkipli, M.F., Inayat, Z., Khan, S., Anthony, B., Chang, V.: From intrusion detection to an intrusion response system: fundamentals, requirements, and future directions. Algorithms **10**(2) (2017). https://doi.org/10.3390/a10020039
7. Bhatele, K.R., Shrivastava, H., Kumari, N.: The role of artificial intelligence in cyber security. August 2020, 170–192 (2019). https://doi.org/10.4018/978-1-5225-8241-0.ch009
8. Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., Dafoe, A., et al.: The malicious use of artificial intelligence: forecasting, prevention, and mitigation (2018). https://doi.org/10.17863/CAM.22520. https://www.researchgate.net/publication/323302750_The_Malicious_Use_of_Artificial_Intelligence_Forecasting_Prevention_and_Mon
9. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun. Surv. Tutor. **18**(2), 1153–1176 (2016). https://doi.org/10.1109/COMST.2015.2494502
10. Chaudhary, H., Detroja, A., Prajapati, P., Shah, P.: A review of various challenges in cybersecurity using artificial intelligence. In: Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020, 829–836 (2020). https://doi.org/10.1109/ICISS49785.2020.9316003
11. Chowdhury, M., Rahman, A., Islam, R.: Malware analysis and detection using data mining and machine learning classification. In: Abawajy, J., Choo, K.-K.R., Islam, R. (eds.) International Conference on Applications and Techniques in Cyber Security and Intelligence–Applications and Techniques in Cyber Security and Intelligence. Advances in Intelligent Systems and Computing, vol. 580, pp. 266–274. Springer-Verlag London Ltd. (2018). https://doi.org/10.1007/978-3-319-67071-3_33
12. Demertzis, K., Iliadis, L.: A bio-inspired hybrid artificial intelligence framework for cyber security. In: Daras, N., Rassias, M. (eds.) Computation, Cryptography, and Network Security. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18275-9_7
13. Dongmei, Z., Jinxing, L.: Study on network security situation awareness based on particle swarm optimization algorithm. Comput. Ind. Eng. **125**, 764–775 (2018). https://doi.org/10.1016/j.cie.2018.01.006
14. Eian, I.C., Yong, L.K., Li, M., Qi, Y.H., Fatima, Z.: Cyber attacks in the era of COVID-19 and possible solution domains (2020). https://doi.org/10.20944/preprints202009.0630.v1

15. El-latif, A.A., Abd-El-Atty, B., Mehmood, I., Muhammad, K., Venegas-Andraca, S.E., Peng, J.: Quantum-inspired blockchain-based cybersecurity: securing smart edge utilities in IoT-based smart cities. Inf. Process. Manag. **58**, 102549 (2021)
16. Geetha, R., Thilagam, T.: A review on the effectiveness of machine learning and deep learning algorithms for cyber security. Arch. Comput. Meth. Eng. **28**(4), 2861–2879 (2021). https://doi.org/10.1007/s11831-020-09478-2
17. Gupta, B.B., Prajapati, V., Nedjah, N., et al.: Machine learning and smart card based two-factor authentication scheme for preserving anonymity in telecare medical information system (TMIS). Neural Comput. Appl. (2021). https://doi.org/10.1007/s00521-021-06152-x
18. Korkmaz, Y.: Kullanıcı Giriş Sistemlerinde Yapay Sinir Ağları Kullan ı larak Ş ifre Güvenlik Sisteminin Geli ş tirilmesi Developing Password Security System By Using Artificial Neural Networks In User Log In Systems Bilgisayar Mühendisli ğ i , Fatih Sultan Mehm, pp. 1–4 (2016)
19. Kumar, M., Singh, N., Kumar, R., Goel, S., Kumar, K.: Gait recognition based on vision systems: a systematic survey. J. Visual Comm. Image Represent. **75**(August 2020), 103052 (2021). https://doi.org/10.1016/j.jvcir.2021.103052
20. Karthik Narayan, L., Sonu, G., Soukhya, S. M.: Fingerprint recognition and its advanced features. Int. J. Eng. Res. Technol. **9**(04), 424–428 (2020). https://doi.org/10.17577/ijertv9is040393
21. Kaye, J., Whitley, E., Lund, D., et al.: Dynamic consent: a patient interface for twenty-first century research networks. Eur. J. Hum. Genet. **23**, 141–146 (2015). https://doi.org/10.1038/ejhg.2014.71
22. Li, H.U.A.: Cyber security meets artificial intelligence: a survey. Front. Inform. Technol. Electr. Eng. **19**(12), 1462–1474 (2018). https://doi.org/10.1631/FITEE.1800573
23. Li, Z., Zheng, L.: The impact of artificial intelligence on accounting. Haisa, 158–169. https://doi.org/10.2991/icsshe-18.2018.203
24. Malatji, M., Marnewick, A., Solms, S.V.: The impact of artificial intelligence on the human aspects of information and cybersecurity. In: Proceedings of the Twelfth International Symposium on Human Aspects of Information Security & Assurance, HAISA 2018 (2018)
25. Marir, N., Wang, H., Feng, G., Li, B., Jia, M.: Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using Spark. IEEE Access **6**, 59657–59671 (2018). https://doi.org/10.1109/access.2018.2875045
26. Naderpour, M., Lu, J., Zhang, G.: A situation risk awareness approach for process systems safety. Safety Sci. **64**, 173–189 (2014). https://doi.org/10.1016/j.ssci.2013.12.005
27. Naik, B., Mehta, A., Yagnik, H., Shah, M.: The impacts of artificial intelligence techniques in augmentation of cybersecurity: a comprehensive review. Complex Intell. Syst. 0123456789. https://doi.org/10.1007/s40747-021-00494-8
28. Radanliev, P., De Roure, D., Page, K., Nurse, J.R.C., Mantilla Montalvo, R., Santos, O., Maddox L.T., Burnap, P.: Cyber risk at the edge: current and future trends on cyber risk analytics and artificial intelligence in the industrial internet of things and industry 4.0 supply chains. Cybersecurity **3**(1) (2020). https://doi.org/10.1186/s42400-020-00052-8
29. Salhi, D.E., Tari, A., Kechadi, M.T.: Email classification for forensic analysis by information gain technique. Int. J. Softw. Sci. Comput. Intell. **13**(4), 40–53 (2021). https://doi.org/10.4018/ijssci.2021100103
30. Salloum, S.A., Alshurideh, M.: Machine learning and deep learning techniques for cybersecurity : a review, vol. 2. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-44289-7
31. Saravanan, A., Bama, S.S.: A review on cyber security and the fifth generation cyberattacks. Oriental J. Comput. Sci. Technol. **12**(2), 50–56 (2019). https://doi.org/10.13005/ojcst12.02.04
32. Sarker, I.H., Kayes, A.S.M., Badsha, S., et al.: Cybersecurity data science: an overview from machine learning perspective. J. Big Data **7**, 41 (2020). https://doi.org/10.1186/s40537-020-00318-5
33. Sarker, I.H., Furhad, M.H., Nowrozy, R.: AI-driven cybersecurity: an overview, security intelligence modeling and research directions. SN Comput. Sci. **2**(3), 1–18 (2021). https://doi.org/10.1007/s42979-021-00557-0

34. Sedik, A.,Tawalbeh, L.O.A.I., Hammad, M.,Latif, A., EI-Banby, G.M., Khalaf, A.F., Samie, F., LLiyasu, A.M.: Deep learning modalities for biometric alteration detection in 5G networks-based secure smart cities. In: IEEE Access, vol. 9, 94780–94788 (2021). https://doi.org/10.1109/ACCESS.2021.3088341
35. Sharma, G.D., Yadav, A., Chopra, R.: Artificial intelligence and effective governance: a review, critique and research agenda. Sustain. Futures **2**(December 2019), 100004 (2020). https://doi.org/10.1016/j.sftr.2019.100004
36. Shidaganti, G.I., Inamdar, A.S., Rai, S.V., Rajeev, A.M.: SCEF: a model for prevention of DDoS attacks from the cloud. Int. J. Cloud Appl. Comput. **10**(3), 67–80 (2020). https://doi.org/10.4018/ijcac.2020070104
37. Shoufan, A., Taha, B.: Machine learning-based drone detection and classification: state-of-the-art in research. IEEE Access **7**, 138669–138682 (2019). https://doi.org/10.1109/access.2019.2942944
38. Tao, F., Akhtar, M., Jiayuan, Z.: The future of artificial intelligence in cybersecurity: a comprehensive survey. EAI Endorsed Trans. Creative Technol. **8**(28), 170285. https://doi.org/10.4108/eai.7-7-2021.170285
39. Truong, T.C., Diep, Q.B., Zelinka, I.: Artificial intelligence in the cyber domain: offense and defense. Symmetry MDPI AG **12**(3), 410 (2020). https://doi.org/10.3390/sym12030410
40. Ubale, T., Jain, A.K.: Survey on DDoS attack techniques and solutions in software-defined network. In: Gupta, B., Perez, G., Agrawal, D., Gupta, D. (eds.) Handbook of Computer Networks and Cyber Security. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-22277-2_15
41. Ullah, F., Babar, M.A.: Architectural tactics for big data cybersecurity analytics systems: a review. J. Syst. Softw. **151**, 81–118 (2019)
42. Vähäkainu, P., Lehto, M.: Artificial intelligence in the cyber security environment artificial intelligence in the cyber security environment (2019)
43. Wiafe, I., Koranteng, F.N., Obeng, E.N., Assyne, N., Wiafe, A., Gulliver, S.R.: Artificial intelligence for cybersecurity: a systematic mapping of literature. IEEE Access **8**, 146598–146612 (2020). https://doi.org/10.1109/ACCESS.2020.3013145
44. Xu, M., Peng, J., Gupta, B.B., Kang, J., Xiong, Z., Li, Z., EI-Latif, A.A.: Multi-agent federated reinforcement learning for secure incentive mechanism in intelligent cyber-physical systems. In: IEEE Internet of Things Journal (2021). https://doi.org/10.1109/JIOT.2021.3081626
45. Xu, Z., Ray, S., Subramanyan, P., Malik, S.: Malware detection using machine learning based analysis of virtual memory access patterns. In: Proceedings of the Conference on Design, Automation & Test in Europe, Lausanne, Switzerland, 27–31, pp. 169–174 (2017)
46. Ye, Y., Chen, L., Hou, S., et al.: DeepAM: a heterogeneous deep learning framework for intelligent malware detection. Knowl. Inf. Syst. **54**, 265–285 (2018). https://doi.org/10.1007/s10115-017-1058-9
47. Zeng, Y., Gu, H., Wei, W., Guo, Y.: Deep-full-range: a deep learning based network encrypted traffic classification and intrusion detection framework. IEEE Access, 1–1 (2019)
48. Zhang, Z., Ning, H., Shi, F., Farha, F., Xu, Y., Xu, J., Zhang, F., Choo, K.K.R.: Artificial intelligence in cyber security: research advances, challenges, and opportunities. Artif. Intell. Rev. 0123456789 (2021). https://doi.org/10.1007/s10462-021-09976-0

Privacy and Security Concerns in Edge Computing-Based Smart Cities

Ashutosh Kumar, Aditya Upadhyay, Neha Mishra, Srawan Nath, Kalu Ram Yadav, and Gajanand Sharma

Abstract People and urban economies have benefited greatly from smart cities' better quality of life and services. They have complete power over physical objects in real-time and can provide intelligent information to people in areas such as transportation, healthcare, building automation, public safety, smart parking, and traffic systems, as well as smart agriculture. Smart city apps have the ability to collect confidential data. At different levels of the architecture, nevertheless, various security and privacy issues can arise. As a result, it's important to consider these security and privacy concerns when developing and implementing applications. The key applications of smart cities are highlighted in this paper and the major security and privacy problems in the design of smart city applications. It discusses some of the latest protection and privacy strategies for information-centric smart city applications and potential research issues that must be identified for performance enhancement.

Keywords Edge computing · Privacy · Fog computing · IoT · Smart city · Cyber-attacks · Encryption

1 Introduction

The population in urban areas has been increasingly growing in recent decades. According to a United Nations Population Fund survey, cities are home to more than half of the world's population [1]. Due to its strong criteria and the realistic context in an urbanised world, the idea of a smart city has gotten much too much attention from academia and industry [2]. Several cities have started to create their smart city plans to improve residents' standard of living and provide better services.

Many countries with rapidly rising populations are investing heavily in smart city programmes. China, for example, is operating on over 200 initiatives related to the

A. Kumar · A. Upadhyay · N. Mishra · S. Nath · G. Sharma (✉)
JECRC University, Jaipur, Rajasthan, India

K. R. Yadav
Government Polytechnic College, Kota, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_5

smart city paradigm [3]. Smart cities-related innovations are allowing city governments to better control their day-to-day activities in order to make people's lives easier. The infrastructure of smart cities involves a range of devices and integrated networks that support people in several settings, including healthcare, smart transportation, parking management, smart traffic networks, smart farming, and smart homes. Information-centric networking or INC models allows packet delivery to be maintained in insecure environments such as leaf pathology [4, 5]. As a consequence, in smart cities, ICN is a substitute for IP-based networks [6]. The fast expansion of wire-free communication technologies which enables small and cheap objects to link to the Internet, as well as the introduction of several low-cost digital devices like sensors and actuators, have led to an increase in the implementation of the Internet of Things (IoT), In which in daily life actual objects become smart objects. In addition to IP-based methods like the one suggested by Sheng et al. [7] In order to improve IoT life and its Network Stability, ICN solutions can be used. Instead of using an IP host, ICN is defined by the term content and central architecture details [8]. The main concept is to fully overhaul the internet's architecture and make it more generic and straightforward [9]. ICN will support multiple Internet of Things situations and overcome their existing weaknesses by using their advantages to incorporate various technologies in compliance like smart cities and smart homes [10]. It can also be used as a platform to link various objects with security features in IoT environments to offer a variety of programmes. Furthermore, during the age of IoT, the use of ICN would result in a reduction in energy consumption [11]. Cities are getting smarter, which can put people's security and privacy at risk [12]. This is due to the resource-constrained existence of smart city devices that make the system vulnerable to a variety of security threats [13]. These flaws could lead to a slew of cyberattacks in smart cities. Malicious attackers, for example, can generate false data while manipulating sensing data, resulting in the loss of influence over extremely advanced systems [12]. In 2015, in Ukraine 230 000 people lost power as a result of a hacker attack on the smart grid or SG that was sent to them in the type of denial of service (DOS) attack [1]. Several resource-constrained devices in smart cities, including cameras that capture and exchange sensitive data, are susceptible to destructive cybersecurity threats, putting people's security and privacy at risk. The area data gathered and monitored by smart devices can reveal people's private way of living and may even result in economic loss as a result of these cyberattacks [12]. There will be increasing demand for intelligent cities to $1.5 trillion by 2020, according to a survey [14]. In reality, governments are in charge of attracting major investments to realise the vision of smart cities [15]. In the city, to provide real-time data about services including traffic, water, air and so and so on, more than a thousand sensor nodes have been installed [16]. However, processing and analysing such a large volume of confidential information poses a range of problems, as well as questions about how to keep private data secure from unauthorised access [17]. Ten Cloud infrastructure will provide efficient data handling and archiving services in the IoT era and smart cities [18].

However, some problems mobility, position awareness, latency, and protection are all issues that can be addressed using the fog computing model in cloud-based IoT

applications [17, 19–21]. It solves these issues by delivering computing services to consumers at the network's edge, thus reducing latency and improving service efficiency [22]. However, because of the variation between the two that is fog and cloud computing, privacy and security are challenging issues in fog computing. As a result, security responses designed for cloud services are not appropriate for IoT applications services available to users. Cyber-attacks can be dealt with using a variety of cryptographic techniques. These strategies, on the other hand, are not suitable for resource-constrained Internet of Things gadgets in smart cities. Offloading additional security operations to a fog-based node, which can allow data security analysis right at the network's edge, is one solution in this regard [23]. Furthermore, in publish and subscribe schemes, which distribute data to subscriber from the publisher, information is spread out through a channel of hackers who can extract sensitive user data. As brokers seek access to publication tags and subscriber preferences, the system must maintain publication and subscription secrecy in this context [24].

The idea of a smart city, according to IBM, is characterized by these three features: instrumented, interconnected, and intelligent [25].

- Instrumented refers to a city that is encircled by a network of devices like sensors and actuators. As a result, these instruments provide accurate and real-time information to the city's core structures.
- Interconnected because it has a large number of networks that work together for sending data through different sources and locations. The use of a right combination of connected and regulated systems can then be used to create a connection between the physical world and real
- Intelligent means the instrumented and integrated world that uses data from different devices and systems like sensors, to enhance the citizens' standard of living.

Despite the aforementioned advantages of smart cities, the massive number of wireless cameras and sensors that capture and transmit data to various internet connections for data processing raise some security and privacy issues. All of these devices are processing critical data and transmitting it over various networks. In today's intelligent environment, data is the most important commodity for users. All data gets processed via operating systems that have security issues, like infrastructural concerns and cyber threats (e.g. DoS attacks). The quality of extremely advanced systems will decrease in the form of services due to these safety issues. To let this system become safer and user friendly, it is essential to resolve these security challenges [26].

Several survey papers on the protection and safety of edge computing-based smart cities have been released. In the work of Zhang et al.10, for example, protection and privacy in the application smart city are investigated. The investigators have addressed some of the protection and privacy issues that these apps face. The authors of Gharaibehet al. [27] described data protection and privacy strategies and explored the innovations that change smart cities into a reality. Different types of privacy, attackers, and needed sources for attacks in smart cities were discussed in Eckhoff

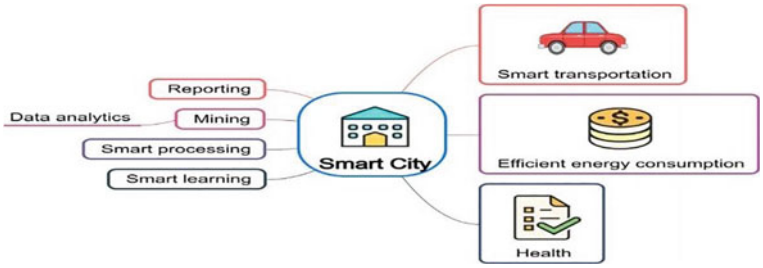


Fig. 1 Features of a smart city [29]

and Wagner's report [28]. The authors have looked at new privacy-enhancing techniques as well as different forms of citizen privacy in smart cities. Similarly, Sicari et al. addressed the key research issues as well as existing security solutions in IoT environments [29] (Fig. 1).

2 The Architecture of Edge-Computing Based Smart Cities

The IoT-based system design focuses on smart city's sensitive issues, is based on the architecture proposed in Hwang and Shin [30]. The following sections address a brief description of all the architectural layers (Fig. 2).

- The physical layer that is at the bottom of the model. It is made up of embedded systems (such as sensing devices) which collect file and transmit them to the architecture's top layer, the network layer, for processing.
- The communication or central layer is the essential layer, also known as the networking layer. Basic networks are required for this layer, including wireless sensor networks (WSNs), the internet, and networking systems. The primary task of the network layer is to relay data from the top layer and to connect multiple entry points such as databases and smart administration.
- The database sheet, also known as the layer formed, works in tandem with the architecture and design upper layers. Application servers and cognitive processing systems form the foundation of this architecture. The primary purpose of this layer is to include intelligent computing strategies such as cloud/edge computing to meet application requirements.
- The virtualization layer, that provides a virtual network process for integrating hardware & software functions into a single logically customizable software-based entity. Network virtualization [31] can necessitate platform and resource virtualization to be effective.
- The analytics of data and mining layer translates the original data to valuable knowledge that improves network stability and provides predictions of system failure. This layer uses a range of data mining and analysis techniques, such as machine learning (ML) algorithms, to analyze data.

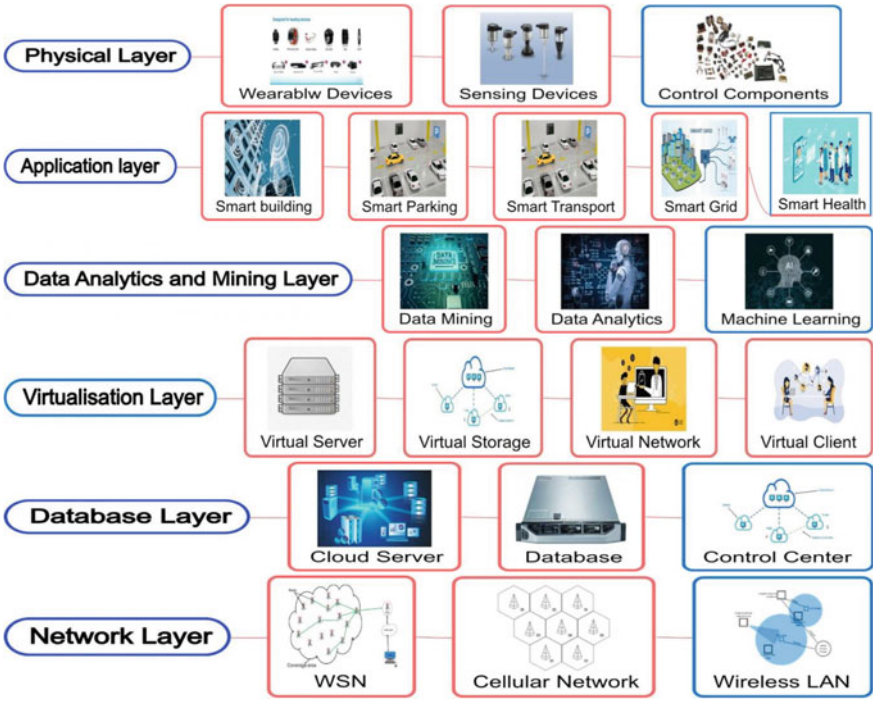


Fig. 2 Layers of edge–computing based smart city [30]

- It is the top layer of a secure IoT architecture and provides users with smart and smart applications and services tailored to their specific requirements. There are some typical examples of these applications. in Table 1.

3 Privacy Concerns in Smart Cities Applications

Major issues have recently arisen in a variety of application scenarios. In smart cities, for example, smart metering infrastructure will regulate residents' private lives, such as their working hours [45]. Furthermore, network operators and device suppliers may have access to sensitive consumer data in smart homes and healthcare [46]. Moreover, smart mobility apps can gather massive data about the individual's movement that can be helpful in determining their mobility pattern and position [47]. With such concerns, because cities are focused on being smarter, privacy and security have now become a critical concern in smart city-related applications. In recent decades, numerous concerning issues are already uncovered in numerous smart home applications [1]. In the parts that follow, we'll mention and briefly address some of the most pressing security concerns.

Table 1 IoT based approach and its application in various aspects of smart life [32, 33]

Aspects	Applications
Environment	Tracking air quality and greenhouse gases, enhancing waste emissions, monitoring eco balance and noise pollution by using technical management devices [32–34]
Health	Providing health services, proper monitoring and diagnosis, identifying diseases on early stage, record keeping and assisting physician by using IHS (Intelligent Health Service) [35, 36]
Transportation	Providing services such as smart vehicle charging, location finding, document verification, parking area navigation so and so forth by using smart transportation system [37–40]
Energy	Monitoring energy consumption, ensuring uninterrupted power supply by deploying energy nodes
Home appliances	Increasing efficiency of energy and home appliances and adding more quality to the livelihood by using smart living devices or automated systems [41]
Electric Grid	Integrating conventional power generations, transmission and distribution sending and receiving data via automated systems by the use of smart electricity grids [42–44]

3.1 Cyber Attacks

Cyberattacks wreak havoc on the security of smart city systems, and they come in two flavours: active and passive. The aim of a passive attack is to use and learn various system database without affecting the system's resources. The key focus of this attack is encrypted data to gain an understanding of the system's configuration and behaviour, as well as its architecture. Since the data is not altered, it is difficult to identify these attacks. As a consequence, it is preferable to concentrate more on avoiding such attacks. Active attacks, but at the other hand, are designed to have an effect or improvement in the system's operation by changing data or injecting inaccurate data into the database. Cyberattacks are mostly motivated by sabotage, deception, and espionage [48].

3.2 Distributed Denial of Service

Bots are one among the most recent problems to pose a significant threat to IoT-based systems. The Mirai botnet, for example, Modifies, deletes, and then spreads infection to other IoT devices on a wide range of computer data, including Router and Webcams, IP cameras, and printers. This may lead to an attack against the server by a distributed denial-of-service [49]. In contrast to many other smart devices such as smartphones and computers, IoT devices are generally built with almost no protection. Since this threat was not recognised until 2016, further research is needed to address it in the future. Otherwise, the IoT ecosystem would be harmed by this

assault [50]. Ghafir et al. proposed a method for preventing such attacks in their paper [51]. To secure critical systems from malware attacks, the authors proposed BotDet, a botnet management & control traffic detection approach. They developed an alert correlation system for polling between the detection algorithms and developed four detection modules in order to identify unorthodox methods used in central botnet commands. The results show that, respectively, the proposed solution blends false and exact rates of over 13% and 82%, which demonstrate the real-time detection capability of the method.

3.3 Unauthorised Data Access

Attackers may use intelligent surveillance systems to capture, distribute, and process personal data such as medical problems, identification and location of individuals in lifestyle and transportation. As a result, smart infrastructures are insecure to data breaches, to address this issue, smart cities will use a variety of security and privacy measures (such as anonymity, authentication, and access controls) to protect sensitive data from hacker attacks [52, 53]. On the other hand, most of the current privacy and protection policies can only protect against malicious attackers and do not take into account the location of possible attackers within [53]. A smart house, for example, may use a surveillance camera to detect fraud or suspicious activity. Internal attackers (employees as well as those with exposure to surveillance records, for example) may obtain personal data or open a door for outside hackers. As a consequence, it is difficult to create a security and privacy scheme in intelligent grids that balances privacy and competitiveness.

3.4 Autonomous Automobiles

Driverless cars, also known as autonomous vehicles, are vehicles that drive themselves without the assistance of a human driver, using a variety of in-vehicle technology and equipment like cameras, Braking devices for global positioning, antilock systems, etc. The development of a safer society with a view both to reducing the number of road accidents and to all of the advantages of this increasingly evolving technology, has gained too much consideration in autonomous vehicles and presents severe privacy and security threats, if it is hacked, threatening citizen data security and life security in smart cities [54]. Elmaghraby and Losavio [54] In various situations, hackers may use safety faults to control the car, such as shutdown of the engine or braking.

Furthermore, the massive amount of personal information gathered from an autonomous vehicle's computer system could result in issues of critical protection and privacy.

4 Privacy and Security Solutions

It is difficult to safeguard the privacy of the inhabitants in smart cities due to the number and complexity of the apps, the fact that each person carries multiple devices, and communications options, and the fact that no organisation, including the city government, is using one technology for more than one function. While a large part of the risk with the exposure of information is found in the application itself, there is another equal portion located in the way that an application utilises the fundamental technologies. Methods of cashless payments (such as smartcards) on public transportation does not poses a privacy risk but can be enacted in a privacy-invasive manner. This section identifies a set of user privacy solutions that are used to address critical threats to data security in smart environments (Fig. 3).

4.1 Encryption

Information privacy is preserved through encryption, which shields messages or data from interception and disclosure. Public-key encryption requires two participants to have an able to share decryption key, the standard symmetric encryption allows encryption, but only associated private keys can decrypt the messages using a public key. Public key encryption is a type of ID encryption where any random string can be a public key. You can encrypt text for a receiver safely using this method, even if a public and private key pair has not been created. Private services that use one's identity are able to discover one's information by means of identity-based encryption. Both the private keys and cipher text are expanded by this type of public-key encryption is an attribute-based encryption depends on the users. If the key attributes for the user are the same as the cipher text attributes, the user is able to decrypt the message. Therefore, increased level of control over encrypted files can be implemented by using this system. where information needs to be encrypted for multiple recipient groups, such as medical personnel with the same sets of attributes,

Fig. 3 Privacy Solutions for Smart Cities [55]

using attribute-based encryption can be employed to include patients, doctors, nurses, and physicians.

4.2 Anonymous or Pseudonymous Credentials

Anonymous or pseudonymous digital credentials It is therefore feasible to give out a set of digital credentials which keep the sender anonymous but give some other attribute, like age or nationality, such as whether the email is fake or not, without making him or her personally identifiable. To be sure, the message has not been changed, an authority has signed messages even if they cannot read the contents Thus, the signature has proven the identity of the sender, but the same doesn't make the sender's identity obvious. use biometric signatures to make and decoy car signing messages to ensure that the origin of the message has been proved, and conceal the identity of vehicle owners [53]. There is no need for a real-name system [55] in a system where users can obtain credentials from anyone they want and where they can use those credentials anonymously. Although an attribute-based authentication method relies on un pseudonymous credentials, it provides anonymous attributions as well access credentials that authenticate the users claims to cloud providers in terms of attributes that are useful in other purposes, for example, allowing users to log in without exposing their identifications. A pseudonymous identity can be established using the public key infrastructure. Each user gets their long-term certificate authority issued, as well as a pool of different short-term ones, anonymous certificates.

4.3 Biometrics

Biometrics are used for authenticating Internet of Things-based infrastructures in almost all of them. Because it heavily depends on human behaviour, this system tends to work on an awful well with bio-data from fingerprints, faces, voices, and signatures, it is considered primitive. It's also one of the most precise, high-performance and efficient methods which also support brain authentication [56, 57], as Amin et al. proposed an additional protocol to safeguard the privacy for data in storage devices, to restrict both mutually untrusted users' abilities to expand their knowledge of that data. Additional data protection measures are outlined in this section should be taken under consideration if bio-based methods haven't been used properly. presented an example in the report of Natgunathan et al. [58] in the section on which it was stated that new biometric methods, like the one described in, should be adopted that would need to secure the privacy of users. Natgunathan et al. [58] not only have potential for general purpose e-commerce and have revealed that such biometrics will have amazing practical use, biometrics can be used for exchanging information with an unmanned aerial vehicle (UAV) and a control station. Sing and Pei et al. [59] provided

an example, which used resources for UAV threat mitigation that are extremely low-cost, as a method of improving safety, by presenting a methodology that can be developed with biometrics. In any case where cybersecurity attacks are relevant, the proposed strategy was found to be applicable is demonstrated to be effective is stated to be in this study: It was shown that the proposed strategy could be used to address issues that affect all sorts of unmanned aerial vehicles [60].

4.4 Data Mining

Whereby assisting in the automation, artificial intelligence is concerned, machine learning is also endeavouring to do everything possible so that systems will be able to learn from experience. Today, current efforts of organisations are investing in machine learning for IoT environments in order to further strengthen their intrusion detection systems [1]. WSN were found in a research study by Alsheikh et al. [61] to demonstrate the benefit of applying ML techniques to increase the protection for an Internet of Things. Luo et al. also developed a machine-driven method to increase data security in WSNs, as was also done by Luo et al. [62] for some complex system vulnerabilities. Another novel model was also proposed by Aminato et al. [63], which involves the use of ML algorithms, to be able to find MAC or phishing attacks in a Wi-Fi network.

4.5 IoT Regulations

One of the critical features of the Internet of Things is the diverse collection of technologies, like (M2M) or machine 2 machine, wireless sensor, and radio-frequency identification (RFID) communication. Although, this same IoT industry is not currently subject to regulation. The outcome of this has had more far-reaching security and privacy ramifications. The application of devices with compromised security in numerous industries, including the armed forces and healthcare, like that of the Internet of Things, is especially valuable, because of the numerous layers is so critical to ensure secure, and because it's so simple to hack [64]. While the industry has been aware of the problems associated with IoT security and privacy, recent attacks have increased awareness of the industry's concerns with internet-connected devices like Jerkins [65] and others have given us another opportunity to showcase how necessary and advantageous security safeguards are, particularly for unintended actions and access to end users' privacy.

5 Suggestions for Secure Structure

Thousands and thousands of resource-constrained systems make up an Internet of things major infrastructure that operate over heterogeneous networks (HetNets). The core elements of a smart city architecture based on IoT are shown. Black networks (BNs), a secure software-defined networking (SDN) controller known as TTP, a centralised registry (UR), and a key management system are among the components (KMS). These four components have different roles in the architecture and are responsible for safe validation and connectivity through HetNets. Data protection, honesty, confidentiality, and authentication are all handled by black networks. The reliable third party is responsible for efficient IoT nodes routing while UR serves to store a database of various devices including nodes, sensors and gateways. Last but not least, KMS is in charge of IoT networks (Fig. 4).

5.1 Black Networks

In an IoT protocol, black networks are used to protect data that includes meta-data linked to each packet. They can protect data using a variety of encryption methods, such as Grain128a or AES in the EAX or OFB modes. At both the connection and network layers, black networks allow identification and secure connectivity. Furthermore, because of the secure messages at the network and connection layer, BNs can minimise a huge spectrum of possible attacks, Provision in IoT-based networks of confidentiality, completeness and privacy.

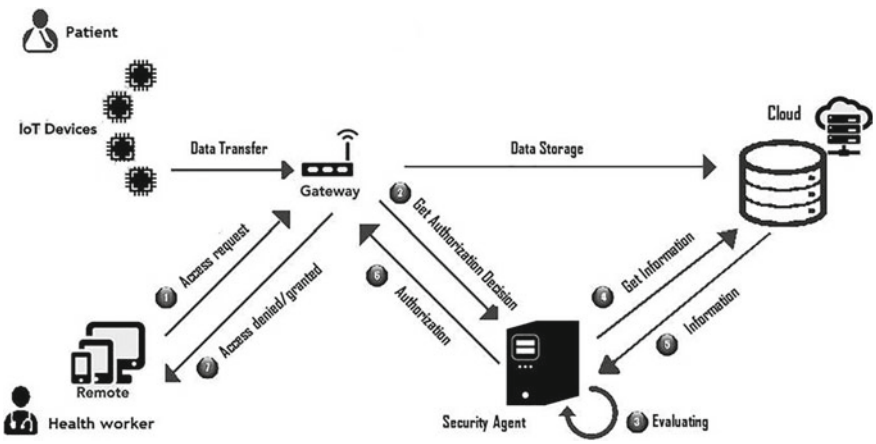


Fig. 4 Layout of a secure architecture [65]

5.2 Software-Defined Networking/SDN Controller

A networking model that offers numerous methods to improve the network security is a network specified by software [66]. In this architecture, a software Defined Networking Controller is used to link network devices with various protocols. Open Flow is the most frequently used protocol to communicate between network devices and the SDN controller. The primary objective of the SDN controller is to tackle the privacy and a security issue faced by IoT-based BNs. Open Flow protocol allows a safe connection to the network devices by the SDN controller. An IoT-based network model with a stable SDN controller is able to cope with sleep and/or wake cycles.

5.3 Unified Registry/UR

The main objective of the UR is to combine various technologies to create intelligent cities IoT infrastructures. The definition for mobile and cross-system IoT nodes can apply to a visiting UR. The consideration of fixed nodes by wireless machines by a variety of IoT networks is important from a security perspective. Wi-Fi is two WLAN network networks that are used by smart cities. Long-term evolution is two. Moreover, a number of protocols like IPv6 via wireless personal area with lower power capacity, ZigBee, and Bluetooth can be used in smart settings as well as a number of methods of addressing such as 128-bit IPv6 addressing, radio wave recovery and 48-bit addressing. All these systems, protocols and schemes require one attribute for identity protection, authentication and authorisation. In addition, wireless communication schemes, protocols and approach addresses need to be translated, and this process is facilitated by UR. The implementation of a UR is difficult due to numerous regulatory, functional and security issues.

An IoT node logical entity can therefore be used in a highly distributed way, which addresses the data and attribute sets of the smart city network.

5.4 System of Key Management

It is the way different working aspects of a cryptographic key for a cryptosystem are treated. It is an important part of any defence infrastructure. Resource restricted devices are used to communicate safely using a symmetrical shared key in IoT settings. Store and use keys securely, should be done. When used on a distributed mobile device, key distribution is a critical issue. In intelligent cities, the hierarchical KMS makes it possible to distribute the keys effectively. It is necessary for IoT device communications to be authenticated. Furthermore, different security protocols and standards are needed for IoT applications because they can deal with different

security vulnerabilities. Message queuing, telemetry transport, and restricted application protocol are some of the most widely used IoT protocols that lack built-in security mechanisms. Constrained application protocols and other protocols, on the other hand, may use protocols like protected sockets layer to improve the security, which means that single-factor authentication is insufficient for safe communication between IoT devices. As a consequence, providing a KMS with a two-factor authentication system is needed to reduce risk and enhance security in IoT networks.

Apart from it, there are various requirements that are needed for the privacy of edge- computing based smart cities that are mentioned in Table 2.

6 Future Challenges and Open Research Directions

A number of strategies have been devised to handle the issue of secure and private data usage in smart cities recently. Though, the implementation for some of these techniques is unrealistic. In urban infrastructures, because of the presence of energy and resource constraints, more-based systems, it is not possible to put in place strong and effective security protocols. Thus, heavier, robust security systems must be examined to ensure they don't impose too much of a burden while still providing a strong level of protection. Although much has been learned about data privacy and security in the last decade, the world still has some unsettled questions to be addressed.

6.1 Rapid Increase in Data

According to IDC, a study from early 2013, the rate of increase in smart devices and big data will lead to greater concern over the problem of cities such applications like intelligent systems that demand higher levels of security and privacy. That is where attackers are able to draw on the human intelligence, obtain or purchase big data, and misuse the information as well as breaching the privacy of the data holders. Many means of detection for attackers are there, for example, methods that use cryptographic techniques such as encrypting fraudulent data or scrambling to avoid detection. As well, it would be useful to make the traceability of the network better so that it could be monitored and controlled more effectively. It is necessary to have data privacy, safety and security to ensure the completeness of the data as well.

6.2 The Adoption of Cloud/fog for Smart Cities

Internet of Things (IoT)-based cloud computing allows devices to keep a huge amount of data locally rather than sending it to a centralised storage location for possible destruction. Even though the process of sending that much data is difficult, with

Table 2 Requirements and applications of edge-computing based smart cities [67]

Requirements	Application
Design privacy	This proposal hopes to deal with the cities that use information as a principal in their security and privacy strategies. This is a new strategy and includes principles that ought to be considered when designing new systems. Specifically, to note, there should be an interventionist approach to resolving incidents of privacy violations rather than waiting for them to happen. However, the design of the system should give special consideration to privacy, and it should be given to be the default
Verification	In order to make sure that these systems meet their security and privacy requirements, this must be taken into consideration. The current testing processes already include privacy-related screening, which means that the issue should be handled by including testing and verification for both the privacy and non-compromising user information. Most significant problem we have with women in Silicon Valley is their unwillingness to smile when they are happy and their inability to stand up when they are angry
Private structure	While multiple privacy safeguards are required for a secure information management, a variety of safeguards are also must be implemented to prevent privacy leaks. For example, the research in the research of Choi et al. had conducted explored a model in which the trustable remote data stores could be relied on, which permits users to connect to trade with a broker, enabling them to broaden access to their personal data
Device access and control	open-source means two or transparent open-source implementations of a more flexible approach to a problem in which implementations may be carried out but neither one's secret inputs nor any external services can be referenced (TTP). Where it is required, a genomic test can be used to improve the overall health planning by expanding upon current methods that use sequencing the results of results in ways that keep the results and genome information private
Secret sharing	This process encourages participants to share information and allows them to communicate in secret. In these experiments, the participants are commonly distributed the equivalent of one share [67]. This procedure has the minimum shares that are required to obtain the secret. As a result, it adds reliability and confidentiality to the system. Secret sharing can be used in two major functions in the city administration: for data storage in metre reading and for distributed measurements, and sensor network data aggregation [68, 69]
Data minimisation	In accordance with the findings of Gürs et al. [70] data minimisation is regarded as one of the most effective privacy measures that companies should adopt. This method could be applied in various ways in an information-rich environment, for example, to look at alternative building design options for an automatic toll pricing, like this [71]. Privacy solutions may also be analysed and/ can also can be utilised in the study of big data for expand on the info. differently to older cities, in response to these older systems, the data should be coded in such a way as to accommodate systems [72]

(continued)

Table 2 (continued)

Requirements	Application
Secure Multiparty devices	It is a public-functioning approach that distributes the calculation of multiple parties, -and-provides for an exacting the amount while concealing all the confidential inputs of all parties, and while relying on nobody in particular (TTP) [73]. Because the test results are not only clinically useful, but also ethically guarded, a health system can use this technique in the structure of only locations in which it is obliged to hold the genetic sequence private [74, 75]

regards to space, frequency, and potency, it must be done because this is extremely valuable. Fog computing enabled them to suggest that IBM apply methods to data in the cloud applications that allow large amounts of data to be processed at the network perimeter instead of relocating that amount of information to the cloud. She was very concerned that he was more concerned about her advancement as a poet than his job as a software engineer. There are a variety of different approaches that smart cities may take to accomplishing sustainable development requirements, including the implementation of smart agriculture, [76] healthcare, [77] water management, and wastewater infrastructure [78].

6.3 Crowd Sensing

Crowd sensing (CS) is a medium for collecting and sharing data on their shared interests through a community of users with devices. Digital gadgets such as Smartphone and other such technology are commonly referred to as mobile CS as surveillance, networking and interaction systems, and crowd sensing (MCS) [79]. Mobile Crowd Sensing can improve the quality of life of people in a variety of fields, including healthcare and transport. Despite all the advantages of MCS, data security and user confidence are serious problems [79]. As a result, these problems in intelligent cities are essential to CS, and must be thoroughly addressed in future studies. In smart cities, MCS already has a broad variety of social and environmental applications. In the climate, MCS is used to measure the level of contamination in a community and the water level in creeks and also to map the natural habitats of wildlife [80]. In addition, users share information through a database server in social MCS to provide a better sense of group problems.

6.4 Confidentiality and Authentication

Authentication is required in smart systems to ensure that only approved users can access services in heterogeneous systems [81]. Smart city IoT devices can authenticate the network, other nodes in the network, other devices and management station messages. Furthermore, since the amount of data that needs to be authenticated is rapidly increasing, it is critical to creating efficient and more advanced technology to help precise and real-time validation in smart cities.

6.5 IoT Networks

The Internet of Things (IoT) can be thought of as HetNets, in which various networks such as mobile networks, social networks, the internet, and relevant factor are interconnected and linked and serve users better. Because of this dynamic world, it is crucial to perform research on successful solutions to address the most recent challenges in smart city security and privacy [60, 82, 83]. The development of successful preventive strategies is critical in this regard. Furthermore, modelling data spread patterns in WSNs is advantageous [84, 85].

6.6 Accessibility and Integrity

Services in smart cities ought to be accessible when they are needed. They should also be able to keep up their effective acts when under attack. Furthermore, a smart device in a smart city should be capable of detecting unusual conditions and preventing further system harm. As a result, robust security techniques must be investigated in order to cope with increasingly sophisticated attacks [86].

Furthermore, the security of IoT devices and the information shared between them and the cloud is crucial [87]. In smart cities, communications take place between different devices; as a result, data can be easily damaged if it is not properly secured during transmission. As a result, it's critical to look at successful methods for ensuring data privacy in IoT system communication in smart cities [88].

7 Conclusion

Smart cities can boost the accessibility of modern societies while also improving people's quality of life and well-being. Cyber Security issues have grown in importance as a result of the introduction of numerous smart systems, necessitating effective and efficient solutions. Furthermore, when designing and implementing new smart

systems, it is critical to understand security and privacy risks. In this paper we have analyzed and discussed safety and privacy issues in smart city information-centered applications. We went through the most common applications for smart cities at the start. The general requirements for smart city services were then provided for security and privacy concerns. We have also proven a number of privacy and safety solutions for a range of applications in intelligent cities that focus on information. The ongoing use of innovation to provide people with networks and interactions and to implement new revolutionary systems and channels, many of which are accessible through the use of smartphones, risks disenfranchising key segments of the population. The rapid pace of technology, which can deliver new and exciting communication methods with healthcare providers, banks, insurance companies, utility providers and transportation companies, is probably an obstacle to adoption among the elderly, who lack the faith and enthusiasm that younger demographic groups are better informed about the technology. Finally, we have discussed some recent research questions to be investigated in detail as the performance of intelligent cities is increased in terms of safety and security.

References

1. Cui, L., Xie, G., Qu, Y., Gao, L., Yang, Y.: Security and privacy in smart cities: challenges and opportunities. IEEE Access. **6**, 46134–46145 (2018)
2. Ever, E., Al-Turjman, F.M., Zahmatkesh, H., Riza, M.: Modelling green HetNets in dynamic ultra-large-scale applications: a case-study forfemtocells in smart-cities. Comput. Netw. **128**, 78–93 (2017)
3. Li, Y., Lin, Y., Geertman, S.: The development of smart cities in China. In: Proceedings of the 14th International Conference on Computersin Urban Planning and Urban Management. Cambridge, MA (2015)
4. Sharma, G., Kumar, A., Saini, A., Upadhyay, A.: Inventors. IoT based smart plant leaf pathology classification and detection using artificial intelligence. India Patent. 202011055879 (2021)
5. Sharma, G., Kumar, A., Upadhyay, A.: Inventors. Development of scene perception system for visually impaired People using IOT Based System. India Patent. 2020111009212 (2021)
6. Wang, M., Wu, J., Li, G., Li, J., Li, Q., Wang, S.: Toward mobility support for information-centric IoV in smart city using fog computing. In: Paper Presented at IEEE International Conference on Smart Energy Grid Engineering (SEGE). Oshawa, Canada (2017)
7. Sheng, Z., Yang, S., Yu, Y., Vasilakos, A.V., McCann, J.A., Leung, K.K.: A survey on the ietf protocol suite for the Internet of Things: standards, challenges, and opportunities. IEEE Wirel. Commun. **20**(6), 91–98 (2013). Vasilakos, A.V., Li, Z., Simon, G., You, W.: Information centric network: research challenges and opportunities. J. Netw. Comput. Appl. **52**, 1–10 (2015)
8. Mars, D., Gammar, S.M., Lahmadi, A., Saidane, L.A.: Using information centric networking in internet of things: a survey. Wirel. Pers. Commun. **105**, 87–103 (2019)
9. Lindgren, A., Ben Abdesslem, F., Ahlgren, B., Schelén, O., Malik, A.: Applicability and trade offs of information-centric networking for efficient IoT (2015)
10. Hahm, O., Baccelli, E., Schmidt, T.C., Wahlisch, M., Adjih, C.: A named data network approach to energy efficiency in IoT. In: Paper presented at: 2016 IEEE Globecom Workshops (GC Wkshps). Washington, DC (2016)
11. Zhang, K., Ni, J., Yang, K., Liang, X., Ren, J., Shen, X.S.: Security and privacy in smart city applications: challenges and solutions. IEEE Commun. Mag. **55**(1), 122–129 (2017)

12. Biswas, K., Muthuk kumarasamy, V.: Securing smart cities using blockchain technology. In: Paper presented at: 2016 IEEE 18th InternationalConference on High Performance Computing and Communications, IEEE 14th International Conference on Smart City, IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS). Sydney, Australia (2016)
13. Perevezentsev, M., Sullivan, F.: Strategic opportunity analysis of the global smart city market. Technical Report. Frost and Sullivan (2013)
14. Albino, V., Berardi, U., Dangelico, R.M.: Smart cities: definitions, dimensions, performance, and initiatives. J. Urban Technol. **22**(1):3–21 (2015)
15. Smart Cities Require Smarter Cybersecurity. Technical Report. http://www.govtech.com/ (2016)
16. He, Y., Yu, F.R., Zhao, N., Leung, V.C., Yin, H.: Software-defined networks with mobile edge computing and caching for smart cities: a big data deep reinforcement learning approach. IEEE Commun. Mag. **55**(12), 31–37 (2017)
17. Maamar, Z., Baker, T., Sellami, M., Asim, M., Ugljanin, E., Faci, N.: Cloud versus edge: who serves the Internet-of-Things better. Internet Technol. Lett. **1**(5), e66 (2018)
18. Kikuchi, Y., Shibata, Y.: Mobile cloud computing for distributed disaster information system in challenged communication environment. In: Proceeding of IEEE 29th International Conferences on Advanced Information Networking and Applications Workshops (WAINA), pp. 512–517 (2015)
19. Mahmud, R., Kotagiri, R., Buyya, R.: Fog computing: a taxonomy, survey and future directions. In: Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives. Singapore: Springer Nature Singapore Pte Ltd, pp. 103–130 (2018)
20. Liang, K., Zhao, L., Chu, X., Chen, H.H.: An integrated architecture for software defined and virtualized radio access networks with fog computing. IEEE Network **31**(1), 80–87 (2017)
21. Chaitanya, K.S., Kumar, A., Kumar, R.: Secure storage of data using cryptography for network. Int. J. Eng. Sci. Technol. **4**(4), 1638–1645 (2012)
22. Abbas, N., Asim, M., Tariq, N., Baker, T., Abbas, S.: A mechanism for securing IoT-enabled applications at the fog layer. J. Sens. Actuator Netw. **8**(1), 16 (2019)
23. Cui, S., Belguith, S., De Alwis, P., Asghar, M.R., Russello, G.: Collusion defender: preserving subscribers' privacy in publish and subscribe systems. IEEE Trans. Dependable Secure Comput. 1 (2019)
24. Ferraz, F.S., Sampaio, C., Ferraz, C.: Towards a smart-city security architecture: proposal and analysis of impact of major smart-city security issues. In: Proceedings of the First International Conference on Advances and Trends in Software Engineering (SOFTENG). Barcelona, Spain (2015)
25. Butt, T.A., Afzaal, M.A.: Security and privacy in smart cities: issues and current solutions. In: Smart Technologies and Innovation for a Sustainable Future: Proceedings of the 1st American University in the Emirates International Research Conference, Dubai, UAE 2017. Cham, Switzerland. Springer Nature Switzerland AG, pp. 317–323 (2019)
26. Gharaibeh, A., Salahuddin, M.A., Hussini, S.J., et al.: Smart cities: a survey on data management, security, and enabling technologies. IEEE Commun. Surv. Tutor. **19**(4), 2456–2501 (2017)
27. Eckhoff, D., Wagner, I.: Privacy in the smart city—applications, technologies, challenges, and solutions. IEEE Commun. Surv. Tutor. **20**(1), 489–516 (2018)
28. Sicari, S., Rizzardi, A., Grieco, L.A., Coen-Porisini, A.: Security, privacy and trust in internet of things: the road ahead. Comput. Netw. **76**, 146–164 (2015)
29. Tan, L., Wang, N.: Future internet: the internet of things. In: Paper presented at: 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE). Chengdu, China (2010)
30. Hwang, I., Shin, D.: Application level network virtualization using selective connection. In: Paper presented at: IEEE International Conference on Consumer Electronics (ICCE). Las Vegas, NV (2018)

31. Anoh, K., Ikpehai, A., Bajovic, D., et al.: Virtual microgrids: a management concept for peer-to-peer energy trading. In: Proceedings of the 2nd ACM International Conference on Future Networks and Distributed Systems (ICFNDS). Amman, Jordan (2018)
32. Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M.: Internet of Things for smart cities. IEEE Internet Things J. **1**(1), 22–32 (2014)
33. Li, X., Lu, R., Liang, X., Shen, X., Chen, J., Lin, X.: Smart community: an internet of things application. IEEE Commun Mag. **49**(11), 68–75 (2011)
34. Ni, J., Lin, X., Zhang, K., Shen, X.: Privacy-preserving real-time navigation system using vehicular crowdsourcing. In: Paper presented 2015 IEEE 84th Vehicular Technology Conference (VTC-Fall). Montreal, Canada (2016)
35. Catarinucci, L., de Donno, D., Mainetti, L., et al.: An IoT-aware architecture for smart healthcare systems. IEEE Internet Things J. **2**(6), 515–526 (2015)
36. Fajardo, J.T.B., Yasumoto, K., Shibata, N., Sun, W., Ito, M.: Disaster information collection with opportunistic communication and message aggregation. J. Inf. Proc. **22**(2), 106–117 (2014)
37. Aouini, I., Azzouz, L.B.: Smart grids cyber security issues and challenges. Int. J. Electron. Commun. Eng. **9**(11), 1263–1269 (2015)
38. Petinrin, J.O., Shaaban, M.: Smart power grid: technologies and applications. In: Paper presented at: 2012 IEEE International Conference onPower and Energy (PECon). Kota Kinabalu, Malaysia (2012)
39. Vlahogianni, E.I., Kepaptsoglou, K., Tsetsos, V., Karlaftis, M.G.: A real-time parking prediction system for smart cities. J. Intell. Trans. Syst. **20**(2), 192–204 (2016)
40. Tang, B., Chen, Z., Hefferman, G., Wei, T., He, H., Yang, Q.: A hierarchical distributed fog computing architecture for big data analysis in smartcities. In: Proceedings of the ASE Big Data and Social Informatics (ASE BD&SI). Kaohsiung, Taiwan (2015)
41. Ding, D., Conti, M., Solanas, A.: A smart health application and its related privacy issues. In: Paper presented at: Smart City Security and Privacy Work shop (SCSP-W). Vienna, Austria (2016)
42. Gungor, V.C., Sahin, D., Kocak, T., et al.: Smart grid technologies: communication technologies and standards. IEEE Trans. Ind. Inform. **7**(4), 529–539 (2011)
43. Bello, O., Zeadally, S.: Toward efficient smartification of the internet of things (IoT) services. Future Gener. Comput. Syst. **92**, 663–673 (2019)
44. Mohanty, S.P., Choppali, U., Kougianos, E.: Everything you wanted to know about smart cities: the Internet of Things is the backbone. IEEE Consumer Electron Mag. **5**(3), 60–70 (2016)
45. Gong, T., Huang, H., Li, P., Zhang, K., Jiang, H.: A medical healthcare system for privacy protection based on IoT. In: Paper Presented 2015 Seventh IEEE International Symposium on Parallel Architectures, Algorithms and Programming (PAAP). Nanjing, China (2015)
46. Ning, Z., Xia, F., Ullah, N., Kong, X., Hu, X.: Vehicular social networks: enabling smart mobility. IEEE Commun. Mag. **55**(5), 16–55 (2017). Wagner, M., Kuba, M., Oeder, A.: Smart grid cyber security: a German perspective. Paper presented at: IEEE International Conference onSmart Grid Technology, Economics and Policies (SG-TEP); Nuremberg, Germany (2012)
47. Delgado-Gomes, V., Martins, J.F., Lima, C., Borza, P.N.: Smart grid security issues. In: Paper presented at: 2015 9th IEEE International Conference on Compatibility and Power Electronics (CPE). Costa da Caparica, Portugal (2015)
48. Zargar, S.T., Joshi, J., Tipper, D.: A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Commun. Surv. Tutor. **15**(4), 2046–2069 (2013)
49. Ghafir, I., Prenosil, V., Hammoudeh, M., et al.: BotDet: a system for real time botnet command and control traffic detection. IEEE Access. **6**, 38947–38958 (2018)
50. Hutson, M.: A matter of trust. Science **358**(6369), 1375–1377 (2017)
51. Weber, R.H.: Internet of things–new security and privacy challenges. Comput. Law Secur. Rev. **26**(1), 23–30 (2010)
52. Chaum, D.: Blind Signatures for Untraceable Payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) Advances in Cryptology, pp. 199–203. Springer, Boston, MA (1983)
53. Schaub, F., Kargl, F., Ma, Z., Weber, M.: V-tokens for conditional pseudonymity in VANETs. In: IEEE Wireless Communication and Networking Conference (WCNC) IEEE Sydney, Australia, pp.1–6 (2010)

54. Elmaghraby, A.S., Losavio, M.M.: Cyber security challenges in smart cities: safety, security and privacy. J. Adv. Res. **5**(4), 491–497 (2014)
55. Zhou, L., Su, C., Chiu, W., Yeh, K.H.: You think, therefore you are: transparent authentication system with brainwave-oriented bio-featuresfor IoT networks. IEEE Trans. Emerg. Top Comput. 1 (2017)
56. Amin, R., Sherratt, R.S., Giri, D., Islam, S.H., Khan, M.K.: A software agent enabled biometric security algorithm for secure file access in consumer storage devices. IEEE Trans. Consum. Electron. **63**(1), 53–61 (2017)
57. Sharma, P., Walia, G.S., Rohilla, R.: Alignment-Free Cancelable Biometric: A Contemporary Survey, Opportunities & Challenges. In: 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, pp. 881–889 (2020). https://doi.org/10.1109/ICISS49785.2020.9316054
58. Natgunanathan, I., Mehmood, A., Xiang, Y., Beliakov, G., Yearwood, J.: Protection of privacy in biometric data. IEEE Access **4**, 880–892 (2016)
59. Singandhupe, A., La, H.M., Feil-Seifer, D.: Reliable security algorithm for drones using individual characteristics from an EEG signal. IEEE Access **6**, 22976–22986 (2018)
60. Sharma, P., Walia, G.S., Rohilla, R.: Recent advancement in cancelable biometric for user recognition: a brief survey. In: 2020 9th International Conference System Modeling and Advancement in Research Trends (SMART), Moradabad, India, pp. 137–146 (2020). https://doi.org/10.1109/SMART50582.2020.9337107
61. Alsheikh, M.A., Lin, S., Niyato, D., Tan, H.P.: Machine learning in wireless sensor networks: algorithms, strategies, and applications. IEEE Commun. Surv. Tutor. **16**(4), 1996–2018 (2014)
62. Luo, X., Zhang, D., Yang, L.T., Liu, J., Chang, X., Ning, H.: A kernel machine-based secure data sensing and fusion scheme in wireless sensornetworks for the cyber-physical systems. Futur. Gener. Comput. Syst. **61**, 85–96 (2016)
63. Aminanto, M.E., Choi, R., Tanuwidjaja, H.C., Yoo, P.D., Kim, K.: Deep abstraction and weighted feature selection for Wi-Fi impersonation detection. IEEE Trans. Inf. Forensics Secur. **13**(3), 621–636. 228–241 (2018)
64. Saleem, J., Hammoudeh, M., Raza, U., Adebisi, B., Ande, R.: IoT standardisation: challenges, perspectives and solution. In: Proceedings of the 2nd ACM International Conference on Future Networks and Distributed Systems (ICFNDS). Amman, Jordan (2018)
65. Jerkins, J.A.: Motivating a market or regulatory solution to IoT insecurity with the Mirai botnet code. In: Paper presented 7th IEEE Annual Computing and Communication Workshop and Conference (CCWC). Las Vegas, NV (2017)
66. Flauzac, O., González, C., Hachani, A., Nolot, F.: SDN based architecture for IoT and improvement of the security. In: Paper presented 201529th IEEE International Conference on Advanced Information Networking and Applications Workshops. Gwangju, South Korea (2015)
67. Wang, Q., Ren, K., Yu, S., Lou, W.: Dependable and secure sensor data storage with dynamic integrity assurance. ACM Trans. Sens. Netw. **8**(1), 9 (2011)
68. Kursawe, K., Danezis, G., Kohlweiss, M.: Privacy-friendly aggregation for the smart-grid. In: Proceedings of the Privacy Enhancing Technologies: 11th Inter-national Symposium, PETS 2011, Waterloo, ON, Canada, July 27–29. Berlin, Germany, Springer, Heidelberg, 175–191 (2011)
69. Shi, J., Zhang, R., Liu, Y., Zhang, Y.: PriSense: privacy-preserving data aggregation in people-centric urban sensing systems. In: Proceedings IEEE INFOCOM. San Diego, CA (2010)
70. Gürses, S., Troncoso, C., Diaz, C.: Engineering privacy by design. Comput. Priv. Data Prot. **14**(3), 25 (2011)
71. Le Métayer D. Privacy by design: a formal framework for the analysis of architectural choices. In: Proceedings of the Third ACMConference on Data and Application Security and Privacy (CODASPY); 2013; San Antonio, TX.
72. Monreale, A., Rinzivillo, S., Pratesi, F., Giannotti, F., Pedreschi, D.: Privacy-by-design in big data analytics and social mining. EPJ Data Sci. **3**(1), 10 (2014)
73. Denning, T., Matuszek, C., Koscher, K., Smith, J.R., Kohno, T.: A spotlight on security and privacy risks with future household robots: attacksand lessons. In: Proceedings of the 11th ACM International Conference on Ubiquitous Computing (UbiComp). Orlando, FL (2009)

74. Jha, S., Kruger, L., Shmatikov, V.: Towards practical privacy for genomic computation. In: Paper presented at: 2008 IEEE Symposium on Security and Privacy (SP). Oakland, CA (2008)
75. Chakrabarty, S., Engels, D.W.: A secure IoT architecture for smart cities. In: Paper presented 2016 13th IEEE Annual Consumer Communications and Networking Conference (CCNC). Las Vegas, NV (2016)
76. Commonwealth Scientific and Industrial Research Organisation (CSIRO).: Phenonet: Distributed Sensor Network for Phenomics Supportedby High Resolution Plant Phenomics Centre, CSIRO ICT Centre, and CSIRO Sensor and Sensor Networks TCP. Technical Report (2011)
77. Zao, J.K., Gan, T.T., You, C.K., et al.: Augmented brain computer interaction based on fog computing and linked data. In: Paper presented 2014 International Conference on Intelligent Environments. Shanghai, China (2014)
78. Mishra, A., Gupta, N., Gupta, B.B.: Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. Telecomm. Syst. 1–16 (2021)
79. He, D., Chan, S., Guizani, M.: User privacy and data trustworthiness in mobile crowd sensing. IEEE Wirel. Commun. **22**(1), 28–34 (2015)
80. Song, H., Srinivasan, R., Sookoor, T., Jeschke, S.: Smart Cities: Foundations, Principles, and Applications. John Wiley and Sons, Hoboken, NJ (2017)
81. He, D., Zeadally, S., Kumar, N., Lee, J.H.: Anonymous authentication for wireless body area networks with provable security. IEEE Syst. J. **11**(4), 2590–2601 (2017)
82. Tewari, A., Gupta, B.B.: Security, privacy and trust of different layers in internet-of-things (IoTs) framework. Futur. Gener. Comput. Syst. **108**, 909–920 (2020)
83. Esposito, C., Ficco, M., Gupta, B.B.: Blockchain-based authentication and authorization for smart city applications. Inf. Proc. Manag. **58**(2), 102468 (2021)
84. Li, D., Deng, L., et al.: A novel CNN based security guaranteed image watermarking generation scenario for smart city applications. Inf. Sci. **479**, 432–447 (2019)
85. Jiang, F., Fu, Y., Gupta, B.B., Liang, Y., Rho, S., Lou, F., Tian, Z.: Deep learning based multi-channel intelligent attack detection for data security. IEEE Trans. Sustain. Comput. **5**(2), 204–212 (2018)
86. Gupta, B.B., Quamara, M.: An overview of Internet of Things (IoT): Architectural aspects, challenges, and protocols. Concurrency and Computation: Practice and Experience, **32**(21), e4946 (2020)
87. Bhushan, K., Gupta, B.B.: Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment. J. Ambient. Intell. Humaniz. Comput. **10**(5), 1985–1997 (2019)
88. Stergiou, C.L., Psannis, K.E., Gupta, B.B.: IoT-based Big Data secure management in the Fog over a 6G Wireless Network. IEEE Int. Things J. (2020)
89. Zhang, K., Lu, R., Liang, X., Qiao, J., Shen, X.S.: PARK: a privacy-preserving aggregation scheme with adaptive key management for smart grid. In: Paper presented at: 2013 IEEE/CIC International Conference on Communications in China (ICCC). Xi'an, China (2013)
90. McGregory, S.: Preparing for the next DDoS attack. Netw. Secur. **2013**(5), 5–6 (2013)
91. Brewster, T.: Smart or stupid: will our cities of the future be easier to hack. The Guardian (2014)
92. Cédric, L.B., Darra, E., Bachlechner, D., et al.: Cyber security for smart cities-an architecture model for public transport. ENISA (2015)
93. Angrishi, K.: Turning the internet of things (IoT) into the internet of vulnerabilities (IoV): IoT botnets. arXiv:1702.03681 (2017)
94. Wang, Y., Wan, J., Guo, J., Cheung, Y.M., Yuen, P.C.: Inference-based similarity search in randomized Montgomery domains for privacy-preserving biometric identification. IEEE Trans. Pattern Anal. Mach. Intell. **40**(7), 1611–1624 (2018)
95. Carlin, A., Hammoudeh, M., Aldabbas, O.: Intrusion detection and countermeasure of virtual cloud systems-state of the art and current challenges. Int. J. Adv. Comput. Sci. Appl. **6**(6) (2015)

96. Shivraj, V.L., Rajan, M.A., Singh, M., Balamuralidhar, P.: One time password authentication scheme based on elliptic curves for Internet of Things (IoT). In: Paper presented 2015 5th National Symposium on Information Technology: Towards New Smart World (NSITNSW). Riyadh, Saudi Arabia (2015)
97. Conti, M., Kumar, M., et al.: Opportunities in Opportunistic Computing. Computer **43**(1), 42–50 (2010)

A Taxonomy of DDoS Defense Mechanism in Software Defined Networking (SDN)

Jasmeen Kaur Chahal, Vidhyotma Gandhi, and Payal Kaushal

Abstract Software-Defined Networking (SDN) have created numerous options in the network technology to differentiate control plane and data- forwarding hardware. This technology has made the network flexible in terms of program editing and dynamic reconfigure. Distributed Denial of Service Attack has become a serious concern for the working of SDN. In this article, we have proposed a taxonomy of DDoS Defense Mechanisms in SDN Environment. We have categorized the various DDoS detection and mitigation techniques with respect to switch intelligence, Defense Deployment, Defense Activity and Network Flow Activities. Also, we have discussed the research issues related to SDN and its security.

Keywords Software defined networking · Distributed denial of service attack · DDoS defense mechanisms · Taxonomy

1 Introduction

SDN has been claimed to be a revolutionizing paradigm shift in networking that give assurance to make the networks programmable, flexible, and dynamic in order to provides a relatively easier management and reduce complexity. The primary concept of SDN is based on the detachment of control plane (the network's control logic) and data plane (network routers and switches). The network switches change to simple forwarding devices responsible for the packet forwarding relies on the rules installed by the centralized controllers. The controllers supervise the whole network; provide feedback with improved decisions as per the global network.

Software-defined Networking Reference Model, consists of three layers, namely, Infrastructure Layer, Control Layer and Application Layer [1]. *Infrastructure Layer (a.k.a. Data Plane)* consists of networking devices (e.g. switches, routers, etc.) that are responsible to form underlying network to forward network traffic. These devices store the information of network status (e.g. network topology, traffic statistics, etc.)

J. K. Chahal (✉) · V. Gandhi · P. Kaushal
Institute of Engineering and Technology, Chitkara University, Punjab, India
e-mail: jasmeen.chahal@chitkara.edu.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_6

and send it to the controller. Also, they process the packets based on the rules given by the controller. *Control Layer (a.k.a. Control Plane)* layer consist of one or more SDN controllers to providing control functionality through APIs (Application Programming Interfaces) to control the whole network. A controller interacts with infrastructure layer using southbound interface (e.g. OpenFlow [2]) and with application layer using northbound interface (e.g. REST API). For large network domain, multiple controllers will exist and these controllers interact with each other using east–west bound interface to coordinate their decision making processes and sharing of network information. *Application Layer* mainly contains of end-user business applications. Examples of such applications include server load balancing, network virtualization, security applications and many more.

2 Taxonomy of DDoS Defense Mechanisms in SDN

Among various threats related to SDN, DDoS attack is the primary threat that can both degrade and disrupt the control plane and data plane causing the denial of network services. Various researchers have developed techniques and methods to minimize the adverse impacts of this attack. Figure 1. shows the division of these techniques according to various features.

(1) **Based on Switch Intelligence**: Many researchers emphasized in adding the intelligence in SDN switches. The major reason behind this is to engage the traffic flows in data plane and to find its processing locality. In this category,

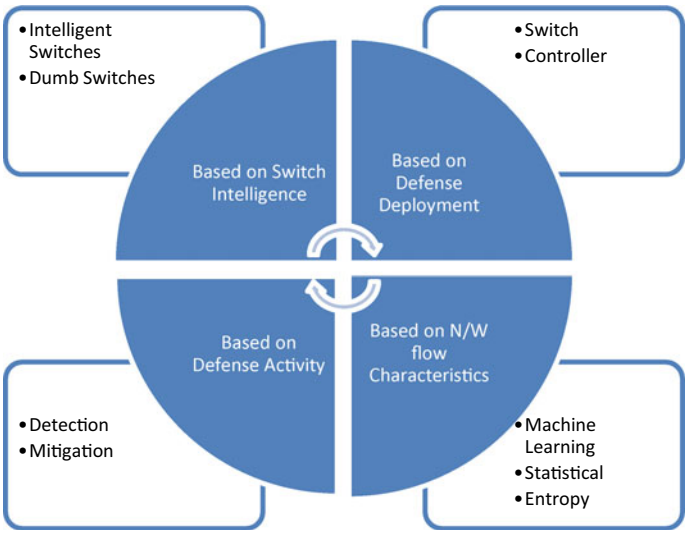


Fig. 1 Taxonomy of DDoS defense mechanism in SDN environment

we have divided the defense mechanism on the basis of intelligent and dumb switches.

(a) ***Intelligent Switches***: Instead of putting whole burden only on controller, many researchers have provided intelligence to the switches [3–7]. The burden of controller get decrease when more number of flows send to it by the switches. It is crucial for switches to make the flow decisions to acquire the safety of the controller. By giving minimal intelligence to the switches the whole SDN paradigm will become more robust and also maintain its uniqueness. On the other hand, the system designer are researching on the consequences of both "pure SDN" paradigm and "capable" switches. The intelligence to the switches may affect the simple objectives of SDN infrastructure.

(b) ***Dumb Switches***: Many of the researchers have implemented their defense mechanisms without providing any amendment in the pure SDN switch.

(2) **Based on Defense Deployment**: It is highly crucial to decide the location of defense technique in SDN paradigm. Accordingly, we have divided the deployment strategy on the basis switch and controller.

(a) ***Switch***: According to the pure SDN arrangement, a switch has to be dumb and should take decisions according to the rules installed by the controller. Many of the researchers have deployed their proposed defense mechanisms on OpenVSwitches [4, 5, 7, 8], so that they can minimize the impact of attack near to its source. However, attackers have different ways to decrease the effectiveness of switches, e.g. overflowing the limited flow table space and overburden the switch's software. Moreover, addition of intelligence to switch's software violates the pure architecture of SDN.

(b) ***Controller***: Controller is the brain of whole network and has global view to detect and mitigate DDoS attack. According to the researchers, the location of defense technique on controller is an ideal location as it has the capability to locate the attack source and it is convenient to put down the switch near to its source [9, 10]. However, controller's sustainability is also on stake as in case of highly flooded attack, it has to respond to numerous rules request which consume its resources such as CPU and storage.

(3) **Based on Defense Activity**: There are many researchers who are giving priority to detect and mitigate DDoS attack. Being a novel architecture, SDN has some features that make it vulnerable to DDoS attack. The fellow researchers are doing efforts to build robust detection mechanism and gave solutions to mitigate the effect of DDoS attack.

(a) **Detection**: Many researchers [9, 11–14] emphasized on the detection of DDOS attack priority basis. It is itself a huge task to find the difference between normal and attack traffic as there are more sophisticated attack strategies such as Layer 7 HTTP attack and Slowloris attack. However,

in SDN, it is extremely important to detect an attack before the complete shutdown of controller.

(b) **Mitigation**: After detection, it is important to decrease the impact of DDoS attack. Researchers have used many techniques such as controller to controller interaction and packet migration [15, 16]. Again, in literature, they have emphasized to empower the controller, as it has a global view, to minimize the attack traffic by deploying flow rules again attack source.

(4) **Based on Network Flow Characteristics**: In this section, we have given a comprehensive review on various techniques used in detection and alleviation of DDoS attack. These solutions work on the basis of detection metric and solutions and depending on these it have been divided into three categories given below:

(a) ***Machine Learning based***: Machine learning algorithms are beneficial solving complicated problems in many fields [12]. These algorithms are also applied for detection of DDoS attacks, and it has been found that they are better than signature-based detection techniques [13]. These ML-based classifiers can be trained to determine the abnormal behaviour of the network traffic with more accuracy. Some commonly used classifiers based on machine learning are Support Vector Machine (SVM), Hidden Markov Model (HMM), Decision Tree (J48), Advanced Support Vector Machine (SVM), Naive Bayes, Logistic regression, Random Trees, Binary Bat algorithm, Random forest, and K-nearest neighbour (KNN). Many fellow researchers used these efficient algorithms for DDoS attack detection, as summarized in Table 1 and discussed below. Machine learning technique provides the ability to learn and improve from experience to take decision without being explicitly programmed. Various machine learning techniques are has been used to detect and mitigate DDoS attacks in SDN environment.

(b) ***Artificial Intelligence Based***: *Neural Network* based detection technique has been used by Cui et al. [17], called SDAnti-DDoS. The detection module is based on BPNN (Back Propagation Neural Network). After analyzing, if the detection module detects the network under DDoS threat, it alerts the traceback module to initiate and find the attack path and source. The attack trajectory and parent information generated by traceback module will be pushed reverse to alleviate attack possibility by blocking the attack traffic.

(c) ***Entropy Based***: Entropy is a measure of uncertainty of an event linked with a desired probability distribution. Very few utilize this method for attack detection. Sahoo et al. [18] use General Entropy (GE) and Generalized Information Distance (GID) utilizes flow table data of switches to find the control layer threat. Destination IP address is used as detection metric. General entropy (GE) outperforms Shannon entropy (Esh) in evaluating DDoS flow from Flash Events (FEs) flow.

Table 1 Summary of various DDoS Defense techniques

S. no	Refs.	Detection/ Mitigation	Algorithm used	Controller	Dataset	Summary
1.	[20]	Mitigation	Statistical based algorithm	–	Traffic Generator tool to generate attack using Mininet	• They have discussed about the DDoS attack scenarios in SDN-based IoT Networks • They have experimented their mitigation technique on testbed • They have shown that random traffic has more impact on network
2.	[21]	Detection	Entropy based	POX	False traffic using Scapy tool	• They have used the traits of SDN to detect DDoS attack • POX controller has been used to carry out the experiments
3.	[22]	Detection	kNN	ONOS	Hibench dataset	• A privacy preserving cross domain attack detection scheme has been proposed • The proposed technique has high efficiency rate
4.	[23]	Detection	Various DL and ML techniques	ONOS	CICDoS2017 and CICDoS2019 datasets	• The experimental setup is using Mininet and ONOS controller • They have compared the results of four Deep Learning (DL) and Machine Learning (ML) • The detection rate is 95% for slow-rate attacks and 98% for high-rate attacks
5.	[24]	Detection & Mitigation	Use IDS and SDN features	–	CAIDA dataset	• Detects DDoS-based cyber-attack scenarios and limits them at their origin at the client side, this way mitigating the negative consequences of the widespread effect of that attack for potential victims

(continued)

Table 1 (continued)

S. no	Refs.	Detection/ Mitigation	Algorithm used	Controller	Dataset	Summary
6.	[25]	Detection	Entropy based	Floodlight	UNB-ISCX, CTU-13 and ISOT datasets	• They have used Weka software for experiment • They have evaluated their proposed technique on high-volume DDoS attacks • They have finalized that high FPR is achieved when there is a dynamic threshold
7.	[13]	Detection	Support Vector Machine (SVM)	Floodlight	Hping3 tool to generate attack using Mininet	• An average detection accuracy and false alarm rate is 95.34% and 1.26% • Since ICMP traffic works without source as well as destination port, results in low detection accuracy of ICMP flood • Generated normal traffic is not comprehensive enough
8.	[9]	Detection	Deep Learning	POX	Traffic generated using a real network and a private network testbed, Hping3	• Deep learning based Stacked Autoencoder (SAE) is used for traffic classification • The SAE model is compared with different models and it outperforms with 95.65% accuracy • Performnace of controller may degrade in large network, due to collection of every packet for feature extraction
9.	[10]	Detection and Mitigation	SVM and SOM	POX	CAIDA dataset	• Combination of SVM and SOM for flow classification and accurate decisions • The whole proposed mechanism is located at controller. Also, policy enforcement module is used to mitigate the attack traffic • The experimental results show higher accuracy rate than the original SVMs and SOM, i.e. 98.13%

(continued)

Table 1 (continued)

S. no	Refs.	Detection/ Mitigation	Algorithm used	Controller	Dataset	Summary
10.	[17]	Detection and Mitigation	Neural Network based BPNN	RYU	D-ITG and TFN2K tool to generate traffic	• Trigger module to chieve rapid response of detection • The results shows that the detection mechansim start quickly and accurately trace the attack source
11.	[8]	Detection and Mitigation	DSOM (Distributed-SOM)	POX	CAIDA and NSL-KDD	• Multiple SOMs are integrated on the OpenFlow switches to resolve the effectiveness at upper and lower layers in complex SDN network • Also, SOM applications are applied to the distributed system as a main classifier at each OpenvSwitch
12.	[26]	Detection	k-Nearest Neighbors (kNN	–	CAIDA "DDoS attacks 2007", CAIDA Anonymized 2008,2000 DARPA LLDOS 1.0 and LLDOS 2.0.2, 1999 DARPA, KDD Cup 1999	• Asymptotic approach of computational security in modern cryptography • The proposed approach helps to determine the malicious traffic on early basis • Extensive evaluation results showed that the proposed technique has ability to detect cross-domain anomalies and preserve the privacy with maximum accuracy
13.	[14]	Detection	SOM and k-NN	POX	DDoS Attack 2007 and CAIDA Anonymized Internet Traces	• Technique reduce the computational time as well as maintain the suitable detection accuracy • Author select the features manually in the proposed techniques
14.	[19]	Detection and Mitigation	Entropy and SVM	POX	False traffic using TFN2K tool	• Detection module is running as an application in the controller • Mitigation based on white-list and dynamic updating of forwarding rules • Number of new source IP increases rapidly as compared to destination IP during attack

(continued)

Table 1 (continued)

S. no	Refs.	Detection/ Mitigation	Algorithm used	Controller	Dataset	Summary
15.	[18]	Detection	General Entropy (GE)	POX	False traffic using Scapy tool	• DestIP is used to detect the attack • Done comparison of Geneal entropy (GE) and shannon entropy (Esh) and concluded that GE outprforms the Esh in detecting DDoS traffic from FE traffic
16.	[4]	Detection and Mitigation	Entropy	OpenDay light	CAIDA	• Middle points are the target instead of end points • Mitigation using Token Bucket Bottle • Accuracy is 100% and False Positive arte is 6.3%
17.	[6]	Detection and Mitigation	Entropy	Ryu	Testbed topology, traffic using tcpreplay	• Provides an intelligence of traffic monitoring to the Switch • Highly effective and accurate in real based scenarios • Has capability of clear monitoring of precision
18.	[12]	Detection	Entropy	POX	False traffic using Scapy tool	• Entropy for DDoS inspection in SDN controller and DDoS attacks was evaluated first time • Average Detection rate is 96% • Technique enable the controller and attack source to detect the attack

Hu et al. (2017) [19] proposed a detection and mitigation module based on **entropy and machine leaning**, against DDoS attack in real time. They collect the traffic information using sFlow and controller based method. The whole FADM framework accurately detects and effectively mitigates the attack traffic in a short time. However, the overhead on controller is high, all modules except mitigation agent, are deployed on controller. Also, the controller handles the normal TCP traffic directly, increases more overhead.

3 Research Issues and Challenges

3.1 Compatibility of SDN with Traditional Network

The centralized nature of SDN makes the management and supervision of network more reliable. However, many companies are afraid of deploying this architecture fully in their network due to various financial and technical issues. The cost will be high if the whole traditional network with simple switches are replaced with SDN enabled switches. Therefore, it is advisable to make a hybrid network (consists of pure and SDN switches), because this not only saves cost but also completes need to merge SDN network with traditional network.

On the contrary, the hybrid network has some issues including handling standard protocols, understanding the difficulty of running multiple architectures and traffic engineering. Also, the management of two different type of switches in one network is very difficult.

3.2 Deployment of DDoS Defense Solutions

As discussed in Sect. 2, most of the researchers have deployed their DDoS detection and mitigation schemes on the controller of the network. Therefore, all switches send queries including legitimate and attack traffic to the controller. Also, controller collects the flow features from SDN switches to monitor whole traffic. This leads to the burden to the communication channel and cause delay in detection. Therefore, many researchers have divided the task of defense on switches and controllers. This distributed way of defense deployment is also a critical area of concern.

3.3 Security Aspects of SDN

The security of SDN architecture has become a critical issue nowadays. The central authority of SDN architecture has make the whole network more reliable, however,

same centralized architecture has some weaknesses that make it unsecure. The attacker overburdens the limited TCAM memory of the switch by sending a large number of attack packets towards it. When this limited switch's memory is becomes full, the switch starts sending the whole packet towards the controller. This not consumes the bandwidth of the communication channel of the controller but also the computational resources of the controller including memory and processing that leads to the delay in response to the normal traffic. Also, there are other attacks such as network manipulation, traffic diversion, traffic sniffing, etc. It is required to save SDN from these threats in order to have working SDN architecture.

3.4 Standardization of Northbound Interface

The northbound interface is a communication path between controller and application plane, but still there is no standard protocol exist for this communication. The controller work according to the applications running on it. It is very easy for controller to install a malicious application on the controller and completely shut down the whole network. So, it is important to give importance to standardize the northbound API.

3.5 Strength Against DDoS Attacks

It is difficult for scientists to completely remove the DDoS attack from the network because to find the attack source from millions of host is impossible. Also, DDoS attack is getting stronger day by day that includes thousands of autonomous systems. Therefore, it is required have a robust DDoS defense mechanism that can maintain the working condition of network while the occurrence of DDoS attack.

4 Conclusion

SDN is a buzzword in the IT world. It has become a promising solution to the many problems of traditional networks. However, there is a lot of research and efforts are required to make it secure. In this paper, we have presented the effect of DDoS attack on SDN paradigm. The taxonomy of DDoS defense mechanisms in SDN environment has been explained in this according to the switch intelligence, defense deployment, defense activity and flow characteristics. A detailed comparison of different detection and mitigation techniques has been shown in the paper. Also, some research gaps have been discussed that requires an immediate attention from research community. For DDoS attack, there is a strong need of improvement in SDN security system in order to deal with.

References

1. Xia, W., Wen, Y., Foh, C.H., Niyato, D., Xie, H.: A Survey on software-defined networking. IEEE Commun. Surv. Tutorials **17**(1), 27–51 (2015). https://doi.org/10.1007/978-3-319-28430-9_9
2. OpenFlow. Open Networking Foundation (ONF). https://www.opennetworking.org/. Accessed 01 March 2018
3. Gkountis, C., Taha, M., Lloret, J., Kambourakis, G.: Lightweight algorithm for protecting SDN controller against DDoS attacks. In: 2017 10th IFIP Wireless Mobile Network Conferences, pp. 1–6 (2017). https://doi.org/10.1109/WMNC.2017.8248858
4. Xu, T., Gao, D., Dong, P., Foh, C.H., Zhang, H.: Mitigating the table-overflow attack in software-defined networking. IEEE Trans. Netw. Serv. Manag. **14**(4), 1086–1097 (2017). https://doi.org/10.1109/TNSM.2017.2758796
5. Pascoal, T.A., Dantas, Y.G., Fonseca, I.E., Nigam, V.: Slow TCAM exhaustion DDoS attack. In: IFIP International Conferences ICT System Security Privacy Protection, pp. 17–31. Springer, Cham (2017). https://pdfs.semanticscholar.org/89c2/bab668f1534fab0453bb5af8373591ec188a.pdf. Accessed 18 April 2018
6. Boite, J., Nardin, P-A., Rebecchi, F., Bouet, M., Conan, V.: StateSec: stateful monitoring for DDoS protection in software defined networks. In: IEEE Conferences Network Softwarization (NetSoft), pp. 1–9 (2017). https://hal.archives-ouvertes.fr/hal-01511012/document. Accessed 18 April 2018
7. Afek, Y., Bremler-Barr, A., Shafir, L.: Network anti-spoofing with sdn data plane. In: INFOCOM 2017-IEEE Conferences Computer Communication, pp. 1–9 (2017). http://www.deepness-lab.org/pubs/infocom17_spoofing.pdf. Accessed 18 April 2018
8. Phan, T.V., Bao, N.K., Park, M.: Distributed-SOM: a novel performance bottleneck handler for large-sized software- defined networks under flooding attacks. J. Netw. Comput. Appl. **91**, 14–25 (2017). https://doi.org/10.1016/j.jnca.2017.04.016
9. Niyaz, Q., Sun, W., Javaid, A.J.: A deep learning based DDoS detection system in software-defined networking (SDN) (2016). arXiv:1611.07400
10. Phan, T.V., Bao, N.K., Park, M.: A novel hybrid flow-based handler with DDoS attacks in software-defined networking. In: IEEE Conference on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pp. 350–357 (2016). https://doi.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.12
11. Mohammadi, R., Javidan, R., Conti, M.: SLICOTS: an SDN-based lightweight countermeasure for TCP SYN flooding attacks. IEEE Trans. Netw. Serv. Manag. **14**(2), 487–497 (2017). https://doi.org/10.1109/TNSM.2017.2701549
12. Mousavi, S.M., St-hilaire, M.: Early detection of DDoS attacks against SDN controllers. In: IEEE International Conference on Computing, Networking and Communications, Communications and Information Security Symposium, pp. 77–81 (2015)
13. Ye, J., Cheng, X., Zhu, J., Feng, L., Song, L.: A DDoS attack detection method based on SVM in software defined network. Secur. Commun. Networks (2018). https://doi.org/10.1155/2018/9804061
14. Nam, T.M., et al.: Self organizing map-based approaches in DDoS flooding detection using SDN. In: 2018 International Conference on Information Networking (ICOIN), pp. 249–254 (2018)
15. Sahay, R., Blanc, G., Zhang, Z., Debar, H.: ArOMA: an SDN based autonomic DDoS mitigation framework. Comput. Secur. **70**, 482–499 (2017). https://doi.org/10.1016/j.cose.2017.07.008
16. Wang, H., Xu, L., Gu, G.: FloodGuard : a DoS attack prevention extension in software-defined networks (2016). https://doi.org/10.1109/DSN.2015.27
17. Cui, Y., et al.: SD-Anti-DDoS: fast and efficient DDoS defense in software-defined networks. J. Netw. Comput. Appl. **68**, 65–79 (2016). https://doi.org/10.1016/j.jnca.2016.04.005

18. Sahoo, K.S., Tiwary, M., Sahoo, B.: Detection of high rate DDoS attack from flash events using information metrics in software defined networks. In: 2018 10th International Conference on Communication Systems & Networks (COMSNETS), pp. 421–424 (2018). https://doi.org/10.1109/COMSNETS.2018.8328233
19. Hu, D., Hong, P., Chen, Y.: FADM: ddos flooding attack detection and mitigation system in software-defined networking. In: EEE Global Communications Conference, pp. 1–7 (2017). https://doi.org/10.1109/GLOCOM.2017.8254023
20. Wang, S., Gomez, K., Sithamparanathan, K., Asghar, M.R., Russello, G., Zanna, P.: Mitigating ddos attacks in sdn-based iot networks leveraging secure control and data plane algorithm. Appl. Sci. **11**(3), 1–27 (2021). https://doi.org/10.3390/app11030929
21. Mishra, A., Gupta, B.B., Perakovic, D., Yamaguchi, S., Hsu, C.H.: Entropy based defensive Mechanism against DDoS attack in SDN-Cloud enabled online social networks (2021). https://doi.org/10.1109/ICCE50685.2021.9427772
22. Dinh, P.T., Park, M.: BDF-SDN: a big data framework for DDoS attack detection in large-scale SDN-based cloud (2021). https://doi.org/10.1109/DSC49826.2021.9346269
23. Yungaicela-Naula, N.M., Vargas-Rosales, C., Perez-Diaz, J.A.: SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning. IEEE Access **9**, 1–1 (2021). https://doi.org/10.1109/access.2021.3101650
24. Manso, P., Moura, J., Serrão, C.: SDN-based intrusion detection system for early detection and mitigation of DDoS attacks. Information **10**(3), 1–17 (2019). https://doi.org/10.3390/info10030106
25. Banitalebi Dehkordi, A., Soltanaghaei, M.R., Boroujeni, F.Z.: The DDoS Attacks Detection Through Machine Learning and Statistical Methods in SDN, vol. 77, no. 3. Springer, US (2021)
26. Zhu, L., Tang, X., Shen, M., Du, X., Guizani, M.: Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks. IEEE J. Sel. Areas Commun. **36**(3), 628–643 (2018). https://doi.org/10.1109/JSAC.2018.2815442

Developing Instrument for Investigation of Blockchain Technology

Dmitry Kushnir, Maxim Kovtsur, Ammar Muthanna, Anastasiia Kistruga, Mark Akilov, and Anton Batalov

Abstract Modern telecommunication technology provides new opportunities for further development of the Smart City operation systems. Smart City systems basically include the Internet of Things (IoT) and CyberPhysical Systems (CPS), which can be implemented by artificial intelligence-based technologies. Significant amount of work about the implementation schemes for the Smart City are using blockchain technology, and the quality of the developed Smart City system strongly depends on the reliability of the blockchain. This article presents an approach to study several blockchain formation aspects. The main goal of this work is to create a research tool for blockchain technology and analyze its capabilities. Such an analysis allows us to determine the scope of possible use of a blockchain, depending on the underlying construction methods, in various fields of application, such as the IoT/CPS. To perform this analysis, several aspects that are needed to explain the basic principles of blockchain formation have been selected. Some minor tasks are specially simplified to avoid distraction from the main ones. This paper also demonstrates the main feature of a blockchain to place information in a distributed ledger. A blockchain formation system has been studied in a specific implementation. A model of a multi nodes system has been constructed, and an analysis of the functioning of a number of software solutions in the field of research modeling of blockchain construction is carried out.

Keywords Smart city · IoT · CPS · Distributed ledger · Blockchain · OpenSSL · SQLite

1 Introduction

Modern achievements of artificial intelligence and recent achievements in the field of robotics and telecommunications provide an opportunity in further development and implementation of the Smart City systems and the services critically important for the

D. Kushnir (✉) · M. Kovtsur · A. Muthanna · A. Kistruga · M. Akilov · A. Batalov
M. A. Bonch-Bruevich Saint-Petersburg State University of Telecommunications, Saint-Petersburg, Russia

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_7

Smart City operation. New opportunities in the application of artificial intelligence to IoT/CPS systems tasks have been discussed in numerous papers during last years [1–9].

One of the most important features of the Smart City system is the security of all the operations including implementation of authentication and authorization, provision of reliability and availability of the provided services. Due to the distributed nature of objects of the Smart City systems, these tasks can be resolved with use of the blockchain technology [10–17]. Hence, security of the blockchain operations is a crucial point for these systems.

Blockchain originally appeared as a distributed registry of the Bitcoin system [18]. Such a register allows to solve the problem of double spending of cryptocurrency. One of the key features of the blockchain is the immutability of the distributed registry. This property allows exchange of data between interested parties, ensuring trust among the initially distrustful parties of information interaction. The rapidly growing success of Bitcoin has attracted a lot of attention of researchers to the new technology underlying it, the blockchain. Despite the fact that the blockchain originated as an infrastructure for cryptocurrency, it has become a technology of distributed systems, which has led to a shift in emphasis from centralized systems to decentralized ones. A decentralized and open architecture is formed on the basis of the blockchain, since it is implemented in a large number of distributed nodes, each of them contains a copy of cryptographically linked records. Such records are organized into blocks in agreement with some consensus protocols among the blockchain nodes. A cryptographically linked blockchain, together with a distributed consensus protocol, ensures the immutability of the blockchain. The openness and immutability of the blockchain allows anyone to check the history of records, which prevents any attempts to interfere with the stored data and protects the information from being changed after being added to the blocks. Thus, trust among the untrusted parties of information interaction can also be created on the basis of a decentralized architecture. The decentralized, open and unmodifiable nature of the blockchain makes it a transparent, publicly verifiable system. In addition, since records are replicated to many distributed nodes, blockchain architecture allows you to get rid of the problem of a single point of failure. The combination of these properties allows us to consider the blockchain technology as the basis for a wide range of applications. Such applications can be solutions for the Internet of Things (IoT) and Cyber-Physical Systems (CPS) in which the possibility of interaction between a huge number of heterogeneous devices is required. Traditional centralized solutions may face such difficulties as distrust of information exchange nodes to each other, a huge number of interactions, the presence of a single point of failure, and a number of others. If we pay attention to CPS, it is also important to note that many existing industrial networks have only now begun to depart from the standards created in the 70s of the last century, and the search for solutions based on blockchain can potentially effectively solve many problems in this area.

The widespread use of various systems that use or claim the use of blockchains for the implementation of certain tasks creates a new reality in the modern digital world. If the first of well-known blockchain implementations was associated with

the creation of a cryptocurrency [18], then today's projects have gone far beyond this narrow framework. A huge number of projects [19, 20] developed on the basis of long-existing platforms or their own implementations create significant difficulties in classifying certain systems and evaluating potential opportunities. In addition, it makes sense to note that from time to time there are projects that are claimed to be based on a functioning blockchain or any other mechanisms with similar functionality, but do not have appropriate technology. Such projects try to exploit this problem and attract potential customers to themselves, diverting significant resources from real tasks. One of the most famous examples of this kind is a pyramid scheme, which was covered by the pseudo-cryptocurrency Onecoin [21].

In addition, it is important to note that even the presence of a successfully working blockchain does not guarantee success of a project implemented on its basis. Depending on the application, different parameters may be required from the blockchain. These parameters can include amount of data that can be written to blocks, speed and delay during recording. High speed of reaction to events, in particular, is necessary for many solutions in IoT and in CPS, for example, in the Internet of Vehicles (IoV). It is also necessary to keep in mind that in some cases the transition to the blockchain or an unsuccessful choice of its specific implementation can only increase overhead costs and efficiency of interaction within the system. This study demonstrates that small improvements to traditional solutions could more effectively overcome existing problems.

In this regard, one of the tasks facing researchers is the creation and active use of the tools that implement and analyze the main mechanisms underlying the construction of the blockchain. In the future, this developed scheme can be used to make assessments of availability and effectiveness of certain blockchain construction technologies to solve specific problems when building IoT/CPS.

2 Problematics

Solutions based on blockchain technologies are developing in new directions and provide new approaches in solutions to both traditional and modern problems. Thus, a wide community of researchers is working on a concept of 'a smart city' which is supposed to solve social problems with the use of telecommunication technologies. One of the tasks for 'a smart city' is an approach to solve a public intercity transportation problem based on a system where participants (passengers, drivers, analytics etc.) are connected with the use of a blockchain, IoT environment and service data (Traffic estimation, Parking service, Navigation services, …) [12].

Another example of one of the most important directions of IoT/CPS is a medical service. Several studies [11, 13], taking into account the distributed nature of interacting elements of medical service located in medical institutions including healthcare providers and patients, propose and analyze the possibility of interaction of healthcare IoT (IoHT), in which the blockchain allows solving the problem of secure

exchange data between system components or provide authentication in the system and ensure the operation of smart contracts.

It should be noted that modern studies develop new important aspects of security problem solving, in particular, they demonstrate such significant advances in quantum computing that it can lead to significant degradation of many other methods underlying security. So, in one of the studies [14], a method of blockchain formation based on the quantum hash function (QHF) is considered to provide a solution to one of the applications to a smart city, the water supply problem. The paper analyzes the addition of information protected by quantum methods to the blocks and substantiates the resistance of this system to several attacks, for example, such as message interception, No-message Attack, Man-in-the-Middle Attack. In general, such solutions allow us to protect data from both an intruder with a traditional computer and a quantum one.

Thus, the construction of various blockchain-based systems has become a noticeable phenomenon in digital data storage systems. In this regard, an important aspect is the analysis of existing and future solutions for the correct implementation of the declared functionality and, in addition, the very fact of the presence of such functionality.

Several studies [15, 16] concentrate on the possibility of the development of various blockchain-based application solutions in the field of IoT and CPS. A feature of such developments is the focus on the already made selection in advance of one, or two, or three specific distributed ledgers for IoT/CPS applications, such as Ethereum, Hyperledger Fabric and IOTA. On the one hand, this approach is justified, since it is impossible to build a solution without taking into account peculiarities of specific components, but on the other hand, such an approach can potentially limit the functionality of the final solution by excluding certain mechanisms of blockchain formation in the solutions being developed.

However, there are studies in the field of building solutions with a choice of blockchain for IoT in Smart City, which develops their own blockchain architecture [17]. The development is justified by the redundancy of the traditional blockchain and the high computational load on individual nodes of the system, which a significant number of smart city elements cannot afford. This approach may be quite appropriate in some applications, but it can significantly complicate the development of a final solution, introduce additional errors into it, and delay implementation.

In this regard, the question of the efficiency of a deliberate selection or development of a blockchain with the necessary characteristics to solve a specific problem remains important. Also, understanding the nature of the functioning of the blockchain is important for all participants in building a complete solution from developers at all levels to service personnel and even users, to maximize the effect be achieved from the implementation of distributed ledgers in various IoT/CPS solutions.

One of the approaches that allow us to solve some of these problems is a demonstration of the implementation of the main functions associated with the formation

of a blockchain. The projects existing in the research area in most cases are ready-made solutions that are difficult to analyze. This applies both to cryptocurrencies and related blockchains [18], as well as to other decentralized systems [22].

There are a number of solutions, both in educational and demonstration applications and for research purposes, which allow analyzing the main stages of block formation, node interaction and consensus building [23–25]. However, all these solutions either simplify the essence of block formation too much, or they work for the user in the form of a black box, which, in fact, without changing the source code, can not be used effectively for research purposes. Table 1 summarizes the main characteristics of the designated solutions in the field of research modeling of blockchain construction with existing gaps. In this paper, we analyze a software model of blockchain formation, which is designed to partially fill these gaps.

The developed tools should have the following functionality:

- Do not require complex configuration and installation;
- Simulate a decentralized distributed system;
- Be able to add nodes arbitrarily;
- Conduct transactions;
- Send transactions to all network participants;
- Check transactions received from other network participants;
- Support a consensus-building mechanism;
- Create blocks;
- Send blocks for verification;
- Check blocks, including displaying their parameters;
- Add blocks to the database, if the check is successful;
- Have a graphical representation for clarity of work.

Table 1 Characteristics of solutions in the field of research modeling of blockchain construction (√- yes, X-no,—partially implemented or requires code editing for analysis)

Solutions	Config/Install required	Ability to add nodes arbitrarily	Implementation of the consensus-building mechanism	Viewing block parameters
Blockchain demo 2.0	X	√	X	–
Visual demo of blockchain technology	X	X	–	√
Building a blockchain by Daniel van Flyman	√	√	–	–

3 Materials and Methods

Various development/programming environments can be chosen to study the methods of building a blockchain. According to the TIOBE Programming Community index for August 2021—C, Python, Java and C++ are the most popular languages. Python is widely used for research and automation, but it has lower performance compared with C++. Additionally C++ is more simple to realize object-oriented programming and support of graphical user interface (GUI), than using C language. C++ needs less computer memory compared with Java to perform the same task. But computer memory consumption is important for IoT and CPS, where devices have limited resources and should provide low energy consumption. So from the point of view of development efficiency and a number of requirements for speed indicators due to the implementation of different nodes of a distributed network on one researcher's computing device, the choice of C++ with the cross-platform Qt framework looks more preferable. For this study, the task of ensuring cross-platform compatibility is important to expand the use of the proposed methods to various platforms.

Since the study of the main stages of building a blockchain requires the implementation of cryptographic functions, a library in which there would be functionality for working with cryptography is needed. OpenSSL is a universal cryptographic library. It supports almost all low-level hashing, encryption and electronic signature algorithms, and also implements the most popular cryptographic standards. OpenSSL is written in the C programming language, which allows it to be used in C++ projects and guarantees high performance and speed of work.

For research purposes, the constructed blockchain assumes that each node stores blocks, both created independently and received from other participants (in this analysis, each node is assumed to be complete, i.e., it stores the entire blockchain). In this case, the research is supposed to be carried out within the framework of a single computing device, and all nodes are formed on the same computer (see Fig. 1). This condition allows us to approach the choice of a data storage system for the blockchain on each node using an embedded cross-platform Database Management System (DBMS), such as SQLite. Choosing this solution for data storage allows us

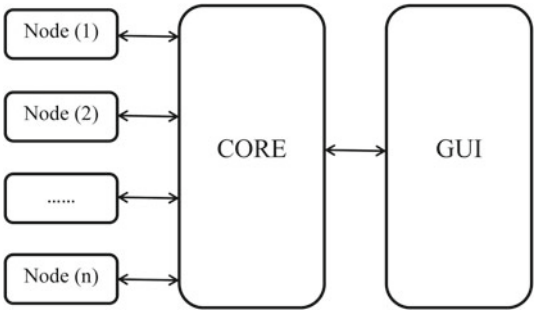


Fig. 1 Implementation of all network nodes within the framework of the developed toolkit

to potentially improve the performance of the final solution, in particular, due to the absence of a client–server architecture, which is not required in this case.

4 Methodology for Analyzing Blockchain Model

As a part of the study, it is necessary to determine the functions that will be analyzed here [26, 27]. The corresponding functions are shown in Fig. 2.

The necessary functions within the framework of the developed solution include:

- Forming nodes:
  - generating a secret key;
  - calculating the public key;
  - formation of the node address.
- Generating data for writing to blocks:
  - preparation of data itself;
  - checking the correctness/balance;
  - creating a digital signature.
- Distribution of the prepared data to all nodes.
- Checking the received data from other nodes.
- Forming a block:
  - collecting data from other nodes in the preliminary version of the block;
  - building a Merkle tree for the block data;

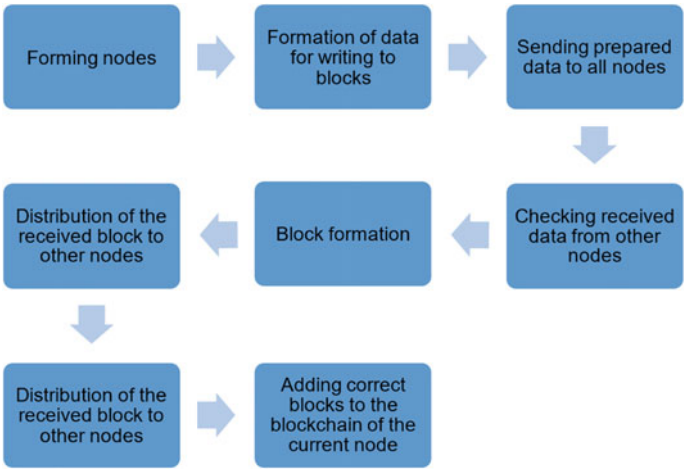

Forming nodes
Formation of data for writing to blocks
Sending prepared data to all nodes
Checking received data from other nodes
Block formation
Distribution of the received block to other nodes
Distribution of the received block to other nodes
Adding correct blocks to the blockchain of the current node

Fig. 2 The main stages of modeling a decentralized distributed system

  - adding service information to a block, including the hash of the previous block;
  - selecting the nonce value to satisfy the condition for the hash value at the current complexity parameter.

- Distribution of the received block to other nodes.
- Checking the blocks received from other nodes.
- Adding correct blocks to the blockchain of the current node.
  Blocks are generated based on the following condition:
  SHA256(SHA256(prvblck), Merkle Root, tmstmp, once) < Difficulty, where:

  - SHA256(prvblck)—hash SHA256 from previous block
  - Merkle Root—hash tree for data in block
  - tmstmp—timestamp for this block
  - once—selectable parameter
  - Difficulty—a parameter that determines the average block generation time

The nodes will be responsible for the main functionality, i.e., work with the formation of data for writing to blocks (transactions), blocks and the database. Each node will have its own database to demonstrate that the network is coming to a consensus. When checking, the databases must be the same. SQLite database is used to store the data on each node. The structure of this database is shown in Table 2. consists of five tables. For example, the "Blocks" table stores info about blocks, known to nodes. "Block_ID" is the block identifier, "PreviousHash" and "CurrentHash" columns show the value of the hash before and after calculations. "Nonce" corresponds to "Nonce", used while calculation of the current block, "MerkleRoot" stores data for Merkle Tree calculation. "time" shows when the block was generated.

The nodes and the program core will model a decentralized distributed system [28], the nodes will communicate with each other using the core. In this case, the kernel can be represented as a data transfer medium between network nodes.

The control will be carried out using a graphical interface. Control signals will be sent to the program core, the core will either redirect the signal to the nodes, or perform other necessary actions.

The graphical interface has the form shown in Fig. 3.

The interface has the following elements:

1. "Add Node" button—adds a new node;
2. The "Start" button—when pressed, all the added nodes begin to continuously calculate blocks;
3. The "Stop" button—stops the calculation of blocks;
4. The "Accept" button—applies the hash complexity specified in (5) for the block;
5. The widget for entering the complexity of the calculated block hash is set by a number from 2 to 8. This number determines the number of zeros at the beginning for a 16-bit hash entry;
6. "Send Tokens"—opens a window for making a transaction;
7. "make block"—when clicked, all nodes start calculating the block hash, but only once. This function is used for step-by-step operation;

Table 2 The structure of the database of the node

Name	Type	Schema
AccountTable		
secretKey	BLOB	"secretKey" BLOB NOT NULL
publicKey	BLOB	"publicKey" BLOB NOT NULL
address	BLOB	"address" BLOB NOT NULL
Blocks		
Block_ID	INTEGER	"Block_ID" INTEGER NOT NULL UNIQUE
PreviousHash	BLOB	"PreviousHash" BLOB NOT NULL UNIQUE
CurrentHash	BLOB	"CurrentHash" BLOB NOT NULL UNIQUE
Nonce	INTEGER	"Nonce" INTEGER NOT NULL
MerkleRoot	BLOB	"MerkleRoot" BLOB NOT NULL
time	TEXT	"time" TEXT NOT NULL UNIQUE
Receivers		
Transaction_H	BLOB	"Transaction_H" BLOB NOT NULL
Receiver_addr	BLOB	"Receiver_addr" BLOB NOT NULL
tokens_count	REAL	"tokens_count" REAL NOT NULL
Senders		
Transaction_H	BLOB	"Transaction_H" BLOB NOT NULL
sender_address	BLOB	"sender_address" BLOB NOT NULL
tokens_count	REAL	"tokens_count" REAL NOT NULL
Transactions		
Block_id	INTEGER	"Block_id" INTEGER NOT NULL
Transaction_H	BLOB	"Transaction_H" BLOB NOT NULL UNIQUE
PublicKey	BLOB	"PublicKey" BLOB NOT NULL
Signature	BLOB	"Signature" BLOB NOT NULL
Summ	REAL	"Summ" REAL NOT NULL

8. The log field of the selected node, displays information for tracking the current operation of this node;
9. Tabs for switching between network nodes;
10. Information about the currently selected node the window for insert data to block shown in Fig. 4. The interface has the following elements:

1. Information about the sender;
2. Field for entering the recipient's address;
3. Input field for the number of tokens to be transferred;
4. Send button;
5. Cancel button.

To implement the program, six classes were created in total, of which three main classes can be distinguished, mainCore which acts as the application core (see Fig. 5),

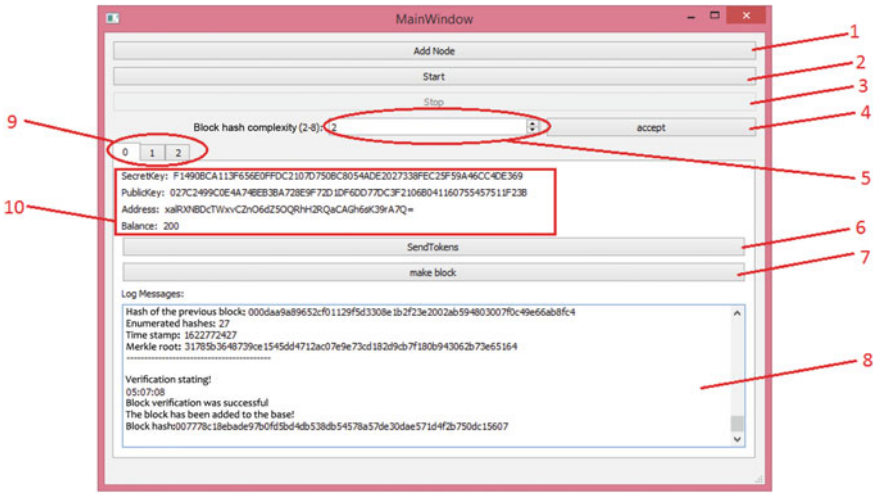
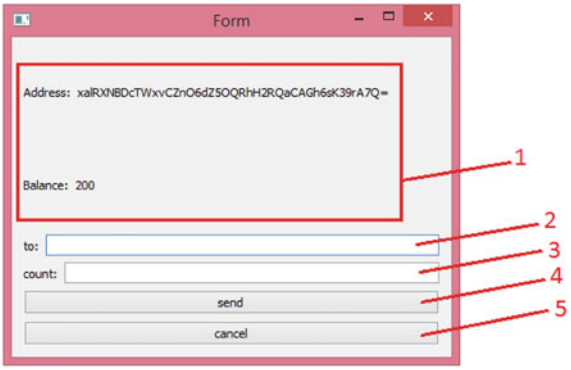


Fig. 3 The main window of the program

Fig. 4 The window for preparing data for entering into the block (performing a transaction)



MainWindow represents the main window (see Fig. 5), and Client implements the node functions (see Fig. 6).

Let's consider some features of the developed tools.

In Qt, the signals and slots technique is used for communication between objects. A signal is emitted, when a certain event occurs. A slot is a function called in response to a certain signal. The signals and slots mechanism is a central feature of Qt and probably it is the part that differs the most from the functions provided by other frameworks.

Signals are emitted by an object when its internal state has changed in any way. Signals are public functions and can be emitted from anywhere, but it is recommended to emit them only from the class that defines the signal and its subclasses.

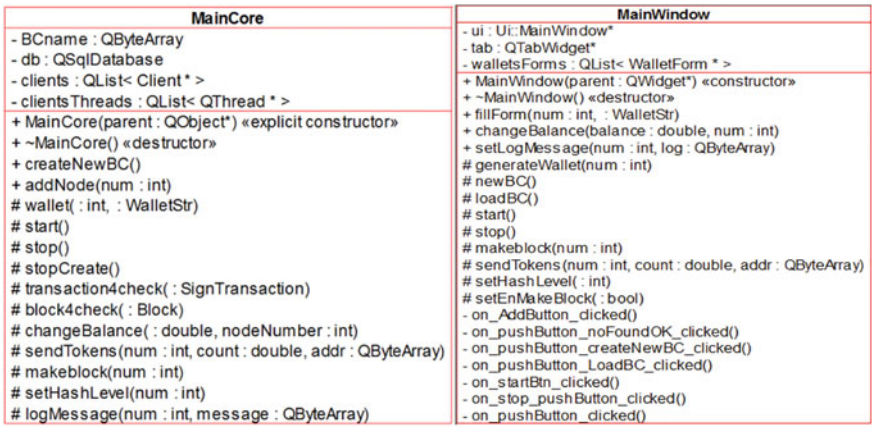


Fig. 5 Diagram of the mainCore class and the MainWindows class

The slot is called when the associated signal is issued. Slots are ordinary C++ functions and can be called in the usual way; their only feature is that signals can be connected to them.

mainCore and MainWindow interact with each other with the following signals: news, generate Wallet, wallet, start, changeBalance, sendTokens, stop, makeblock, setHashLevel, logMessage.

The generateWallet(int num) signal is emitted when the "add node" button is pressed. This signal is associated with the addNode slot(int num) and passes the "int num" parameter—this parameter provides the number of the node to be added, it is needed for interaction between the nodes and the graphical interface. When the generateWallet signal is emitted, the "addNode" slot is executed. Let's look at it in more detail.

"addNode"—creates new nodes. Keys are generated for new nodes, this is where the OpenSSL library is used. The EC_KEY_new_by_curve_name() function allocates memory and binds the EC_KEY object to the specified curve. In our case, the secp256k1 curve is selected. EC_KEY_generate_key() generates a new private and public key. We get the address by taking the SHA-256 hash from the public key.

The QSqlDatabase db object is used to create a database for a new node. The database is created under the name corresponding to the public key of the node.

Next, a new node is created, its initialization is performed using the generated keys. For each new node, a separate thread is created in which it will work. This way will increase performance in multi-core CPU systems.

The Client type object shown in Fig. 7 simulates client actions, and also uses the startBlock slot and the makeBlock function to generate new blocks.

First, a signal is emitted with a log message that the calculation of blocks begins, and then the makeBlock() function is called, shown in Fig. 8.

In this function, the block header is formed, and the creator is initialized by the block header structure and the startCreateHashBlock() signal is emitted.

Client
- stopMakeBlocks : bool
- hashLevel : int
- nodeNumber : int
- balance : double
- secretKey : QByteArray
- publicKey : QByteArray
- address : QByteArray
- TrasactionBuffer : QList< SignTransaction >
- hashThread : QThread*
- creator : HashBlockCreator*
- block4send : Block
- db : QSqlDatabase
- folder : QByteArray
- group : EC_GROUP*
- key : EC_KEY*
+ Client(parent: QObject*) «explicit constructor»
+ init(sk : QByteArray, pk : QByteArray, folderName : QByteArray, numNode : int) : int
+ makeSign(: Transaction) : QByteArray
+ checkTrInBlock(: QList< SignTransaction >) : bool
+ checkTr(: SignTransaction) : bool
+ makeMerkleRoot(: QList< SignTransaction >) : QByteArray
+ makeTransactionHash(: Transaction) : QByteArray
+ makeTransactionHash(: SignTransaction) : QByteArray
+ getHashLastBlock() : QByteArray
+ getBlockID(hash : QByteArray) : qint64
+ getSenderSum(addr : QByteArray) : double
+ getReceiverSum(addr : QByteArray) : double
+ getBalance(addr : QByteArray) : double
+ insertTransactions(: QList< SignTransaction >, : QByteArray)
+ makeBlock()
+ makePackTransactions(): QList< SignTransaction >
# transaction(: SignTransaction)
# block4check(: Block)
# nodeCreated(: int, : WalletStr)
# startCreateHashBlock()
# stopCreateHashBlock()
# blockCreated()
# changeBalance(: double, nodeNumber : int)
# logMessage(num : int, message : QByteArray)
+ checkBlock(: Block)
+ checkTransaction(: SignTransaction)
+ startBlock()
+ stopCreated()
+ stopMake()
+ makeOneBlock(num : int)
+ makeTransaction(num : int, count : double, addrReceiver : QByteArray)
+ setHashLevel(num : int)
- blockHashCreated(hash : QByteArray)

Fig. 6 Diagram of the client class

```
void Client::startBlock()
{
    emit logMessage(nodeNumber,"Start calculate blocks\n");
    stopMakeBlocks = false;
    makeBlock();
}
```

Fig. 7 Client type object

Creator is an object of the HashBlockCreator type shown in Fig. 9. HashBlock-Creator is a class created for selecting a hash of a given complexity. The init() method initializes a new block header. The hash is selected in the start slot, this slot is called, when the start Create Hash Block() signal is triggered.

```
void Client::makeBlock()
{
    block4send.prevBlockHash = getHashLastBlock();
    block4send.transactions = makePackTransactions();//add transactions
    block4send.merkleRoot = makeMerkleRoot(block4send.transactions);
    block4send.nonce = -1;
    headBlockForHash str;
    str.prevBlockHash = block4send.prevBlockHash;
    str.merkleRoot = block4send.merkleRoot;
    str.timeStamp = QString::number(QDateTime::currentSecsSinceEpoch())
                                                                .toUtf8();
    str.nonce = -1;
    creator->init(str);
    emit startCreateHashBlock();
}
```

Fig. 8 makeBlock function

```
void HashBlockCreator::start()
{
    stopCreate = false;
    do{
        QByteArray headArr;
        QDataStream s(&headArr,QIODevice::WriteOnly);
        str.nonce++;
        str.timeStamp = QByteArray::number(QDateTime::
                                            currentSecsSinceEpoch());
        s<<str;
        hash = QCryptographicHash::hash(headArr,
                                QCryptographicHash::Algorithm::Sha256);
        QByteArray checkBytes = hash.toHex();
        checkBytes.remove(8,checkBytes.length());
        levelFlag = true;

        for(int i = 0; i < hashLevel;i++){
            if(checkBytes.at(i)!= '0'){
                levelFlag = false;
            }
        }
    }while((!levelFlag) && (!stopCreate) );
    if(!stopCreate){
        emit createdCorrectHash(hash);
    }
}
```

Fig. 9 HashBlockCreator source code

In this slot, the block header is hashed, the block structure is written to the QByteArray header object, hashing is performed using the QCryptographicHash::hash function, according to the SHA-256 algorithm. The resulting hash is converted to a 16-bit form and checked. If the received hash does not satisfy the specified complexity and there is no signal that it is necessary to stop the hash selection, the actions are repeated, but the fields of the block header structure such as nonce and timeStamp are changed. The nonce field is incremented by 1, timeStamp gets a new timestamp. Otherwise, it is checked whether a stop signal has been received. If not, a createdCorrectHash signal is emitted that a hash of the desired complexity has been selected.

The start slot is executed in a separate thread, so that at the time of hash selection, the node can process transactions and receive signals from other nodes. Additionally, such solutions increase performance by using different cores in modern CPUs

and allow to estimate of resources needed for this action. That's important for the estimation of algorithm performance for IoT and CPS, where CPU resources can be rather limited.

In the process of calculating the block hash, the node continues to interact with other nodes using the MainCore class and receiving information from them about new data that needs to be included in the block. In addition, a node can receive a block already formed by another node. In this case, the node checks the received block, in particular, for the correctness of the data contained in it, for the connection of the received block with the current chain and compliance with the current complexity parameter. Upon successful verification, the received block is included in the blockchain, stored in the node's database to table "Blocks" and is taken as the starting point for calculating the following blocks. In case of any errors during the checks, the received block is ignored.

5 Developed Application of the Analysis Blockchain Technique

The tool for analyzing the main stages of building a blockchain at the first stage forms the necessary number of nodes involved in further research. Figure 10 shows an example of the blockchain with three created nodes where Secret Key, Public Key and Address are generated for each node. A separate SQLite database is created for each node, named according to the node's Public Key. The generated blocks are placed into this database on each node.

The next step is to create new blocks. Blocks can be created both automatically and step-by-step. Blocks are created, validated, and added to the node databases. The network comes to a consensus. This can be seen from the same data in the databases of the nodes.

For the possibility of a detailed study of the performed actions at each step of the simulation, all operations are saved in a log file for further analysis.

The introduction of arbitrary records into the blockchain in this simulation is implemented in the form of creating transactions. Transactions are formed, signed, added to blocks and written to node databases. The balance of nodes changes, tokens come to the destination address (Figs. 11 and 12).

An important parameter for each blockchain system is the average block creation time and possible deviations from the expected value. These parameters can affect the time to reach a network consensus, the necessary time delays for recognition by system participants, and the possibility of certain types of attacks. In some technologies of blockchain formation, the main parameter that affects the formation time of the block creation is the complexity parameter. In the model under study, the complexity is set as the upper bound of the calculated hash value, given as the number of leading zeros in the hexadecimal representation of the boundary, an additional parameter will be the number of nodes involved. The effect of complexity on the time of making

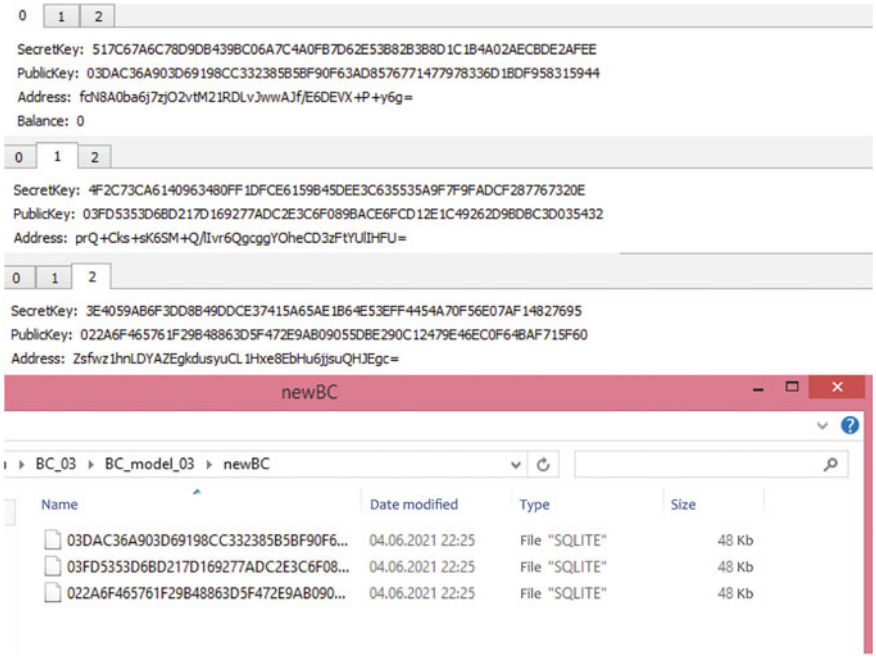


Fig. 10 Adding new nodes. Checking the creation of the corresponding databases

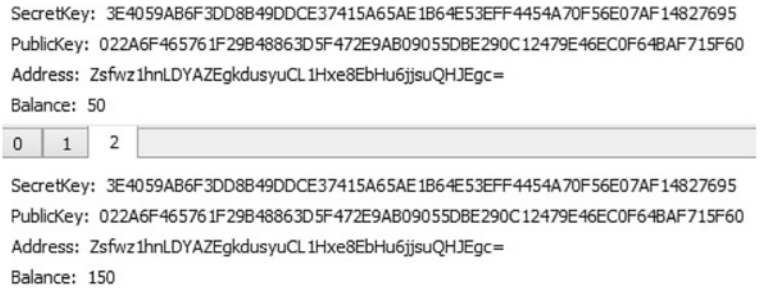


Fig. 11 The balance of node number 2 before and after the formation of a new block

Fig. 12 Transaction records in the database

entries in the distributed registry for a network of four nodes and complexity 6 is shown in Fig. 13, and for complexity 7 in Fig. 14. The dots indicate the time spent on creating the next block in the system.

The graphs clearly show the dependence of the average time spent on creating the next block on the complexity parameter with characteristic outliers corresponding to the Poisson distribution law.

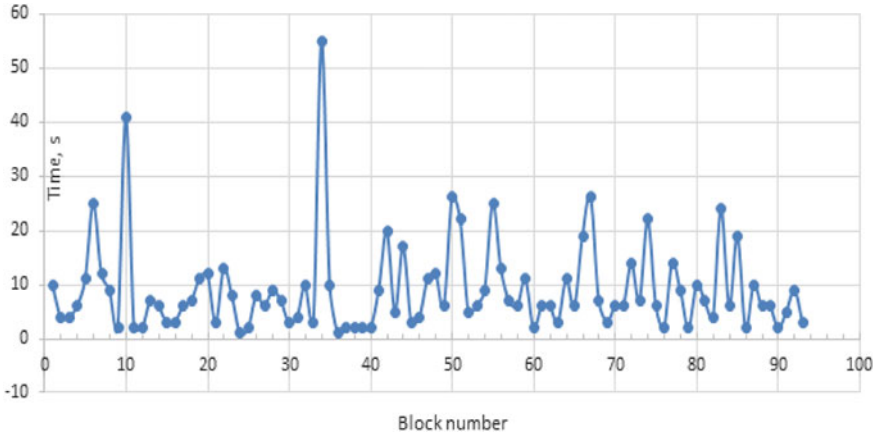


Fig. 13 The time spread during the formation of blocks on difficulty 6

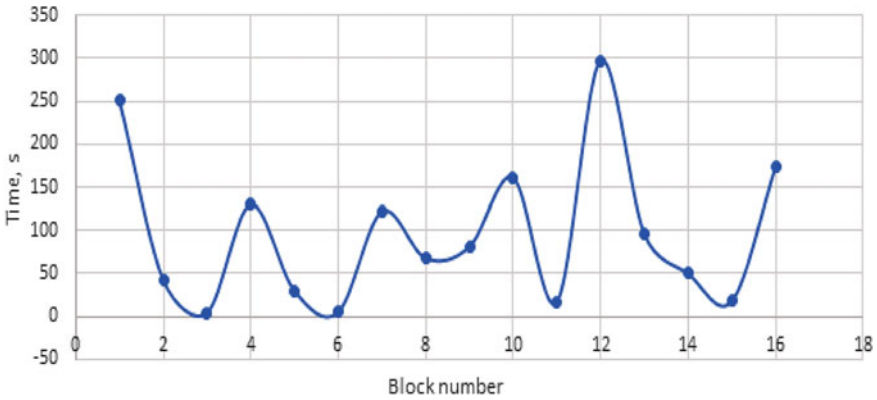


Fig. 14 The time spread during the formation of blocks on difficulty 7

6 Conclusion

The widespread and significant perspectives of IoT/CPS-based systems in Smart Cities require them to have a completely new level of security from all the participants, including end users. In some cases, for example, in IoHT, the requirements for reliability and trust become unprecedented, which demands development of various tools of analysis and modeling, available to a wide range of users.

The analyzed tool used for performing research on blockchain technologies combines easiness of use, ability to track each step of building a blockchain and possibility to check current parameters and obtain aggregating data, which gives certain advantages over similar solutions considered earlier. The presented stages allow us to evaluate capability of traditional approaches and the potential advantages of the latest developments in this area. The paper presents and studies, in particular, such components of this technology as:

- creation of keys to confirm the authenticity of entries in the distributed registry;
- formation of nodes, i.e., participants of a decentralized network;
- managing the network complexity parameter, to influence the average speed of generating blocks in a distributed ledger;
- construction of the Merkle tree, formation of the source block and selection of the nonce parameter to obtain the final block of the system;
- checking the created blocks for compliance with the current system rules by the block creator;
- independent view of the status of the blockchain at each node of the network;
- checking the correctness of the created blocks by other participants of the distributed network.

An important feature of this analysis is the ability to switch the mode from analyzing the general parameters of the system to the step-by-step execution mode and monitoring all the parameters of operations at each completed step. Thus, the results presented in the paper allow, on the one hand, to advance in the field of evaluating the functionality of various implementations of blockchain construction methods, and on the other hand, to solve the problem of detailed analysis and demonstration of the capabilities of the fundamental components of blockchain technology. The result of the analysis of distributed ledger methods and technologies underlying the implementation of the blockchain allows us to choose a solution for specific tasks in IoT/CPS or Smart City and determine the security parameters of the implemented systems. In addition to analyzing the properties of the blockchain scheme implemented in the systems, it is important to analyze different approaches of modeling an interaction between the nodes with a distributed ledger, because properties of IoT/CPS systems will be determined by the whole structure implementation.

References

1. Sarivougioukas, J., Vagelatos, A.: Modeling deep learning neural networks with denotational mathematics in UbiHealth environment. Int. J. Softw. Sci. Comput. Intell. (IJSSCI) **12**(3), 14–27 (2020)
2. Masud, M., Gaba, G.S., Alqahtani, S., Muhammad, G., Gupta, B. B., Kumar, P., Ghoneim, A.: A lightweight and robust secure key establishment protocol for internet of medical things in COVID-19 patients care. IEEE Internet Things J. (2020)
3. Ghoneim, A., Muhammad, G., Amin, S.U., Gupta, B.: Medical image forgery detection for smart healthcare. IEEE Commun. Mag. **56**(4), 33–37 (2018)
4. Sahoo, S.R., Gupta, B.B.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Appl. Soft Comput. **100**, 106983 (2021)
5. Javaid, M., Haleem, A., Singh, R.P., Suman, R.: Artificial intelligence applications for industry 4.0: a literature-based study. J. Ind. Integr. Manage. 1–29 (2021)
6. Elmisery, A.M., Sertovic, M., Gupta, B.B.: Cognitive privacy middleware for deep learning mashup in environmental IoT. IEEE Access **6**, 8029–8041 (2017)
7. AlZu'bi, S., Hawashin, B., Mujahed, M., Jararweh, Y., Gupta, B.B.: An efficient employment of internet of multimedia things in smart and future agriculture. Multimedia Tools Appl. **78**(20), 29581–29605 (2019)
8. Khan, A.A., Kumar, V., Ahmad, M., Gupta, B.B., El-Latif, A., Ahmed, A.: A secure and efficient key agreement framework for critical energy infrastructure using mobile device. Telecommun. Syst. 1–19 (2021)
9. Xu, M., Peng, J., Gupta, B.B., Kang, J., Xiong, Z., Li, Z., Abd El-Latif, A.A.: Multi-agent federated reinforcement learning for secure incentive mechanism in intelligent cyber-physical systems. IEEE Internet Things J. (2021)
10. Gupta, B.B., Li, K.C., Leung, V.C., Psannis, K.E., Yamaguchi, S.: Blockchain-assisted secure fine-grained searchable encryption for a cloud-based healthcare cyber-physical system. IEEE/CAA J. Autom. Sinica (2021)
11. Nguyen, G.N., Le Viet, N.H., Elhoseny, M., Shankar, K., Gupta, B.B., Abd El-Latif, A.A.: Secure blockchain enabled Cyber–physical systems in healthcare using deep belief network with ResNet model. J. Parallel Distrib. Comput. **153**, 150–160 (2021)
12. Abbas, K., Tawalbeh, L.A.A., Rafiq, A., Muthanna, A., Elgendy, I.A., El-Latif, A., Ahmed, A.: Convergence of Blockchain and IoT for secure transportation systems in smart cities. Secur. Commun. Netw. 5597679 (2021)
13. Abou-Nassar, E.M., Iliyasu, A.M., El-Kafrawy, P.M., Song, O.Y., Bashir, A.K., Abd El-Latif, A.A.: DITrust chain: towards blockchain-based trust models for sustainable healthcare IoT systems. IEEE Access **8**, 111223–111238 (2020)
14. Abd El-Latif, A.A., Abd-El-Atty, B., Mehmood, I., Muhammad, K., Venegas-Andraca, S.E., Peng, J.: Quantum-inspired blockchain-based cybersecurity: securing smart edge utilities in IoT-based smart cities. Inf. Process. Manage. **58**(4), 102549 (2021)
15. Lee, J., Azamfar, M., Singh, J.: A blockchain enabled Cyber-Physical System architecture for Industry 4.0 manufacturing systems. Manuf. Lett. **20**, 34–39 (2019)
16. Pustišek, M., Kos, A.: Approaches to front-end IoT application development for the ethereum blockchain. Proc. Comput. Sci. **129**, 410–419 (2018)
17. Dlimi, Z., Ezzati, A., Ben Alla, S.: A lightweight blockchain for IoT in smart city (IoT-SmartChain). CMC-Comput. Mater. Continua **69**(2), 2687–2703 (2021)
18. Satoshi Nakamoto. Bitcoin: a peer-to-peer electronic cash system [Online]. https://bitcoin.org/bitcoin.pdf (2008). Accessed 20 June 2021
19. Tabernakulov, A., Koifmann, J.: Blockchain in practice. Alpina Publisher (2019)
20. Instruction of the Prime Minister Dmitry Medvedev on the possibility of using blockchain technology in the system of public administration and the economy of the Russian Federation on March 6, 2017 [Online]. http://government.ru/orders/selection/401/26653. Accessed 20 June 2021

21. BBC News: Cryptoqueen: how this woman scammed the world, then vanished [Online]. www.bbc.com/news/stories-50435014 (2016). Accessed 20 June 2021
22. Tapscott, D., Tapscott, A.: Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world. Penguin (2016)
23. Blockchain Demo 2.0 [Online]. https://blockchaindemo.io. Accessed 20 June 2021
24. Brownworth, A.: Visual demo of blockchain technology. [Online] http://anders.com/blockchain. Accessed 20 June 2021
25. Van Flymen, D.: "Learn blockchains by building one." The fastest way to learn how Blockchains work is to build one. Apress (2017)
26. Singhal, B., Dhameja, G., Panda, P.S.: Beginning blockchain: a beginner's guide to building blockchain solutions. Apress (2018)
27. Antonopoulos, A.M.: Mastering bitcoin: programming the open blockchain. O'Reilly Media, Inc. (2017)
28. Minyaev, A.A., Krasov, A.V., Saharov, D.V.: The method and methodology of efficiency assessment of protection system of distributed information systems. In: 2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), pp. 291–295. IEEE (2020)

An IoT-Based System for Expert User Supporting to Monitor, Manage and Protect Cultural Heritage Buildings

Mario Casillo, Francesco Colace, Angelo Lorusso, Francesco Marongiu, and Domenico Santaniello

Abstract The advent of the Internet of Things has led us towards the use of systems and infrastructures capable of collecting information through sensors. These technologies allow us to collect data stored in increasingly complex and rich databases that can increase the capacity of analysis and allow a use that leads to increasingly innovative services. This approach could be handy in application to the field of Cultural Heritage. In particular, buildings and artefacts of historical-cultural value need constant monitoring and maintenance. A system that can support expert users to monitor, manage, predict and choose the types of interventions to be performed could be beneficial in preserving Cultural Heritage. The proposed paper aims to present a system that can support expert users in monitoring, management, and maintenance using sensors based on the IoT paradigm. In order to test the proposed system, a prototype has been developed and tested in a real scenario: the monitoring of a building inside the Pompeii Archaeological Park. The preliminary results are promising.

Keywords Internet of things · Cultural heritage · Preventive maintenance · Digital twin · Sensors

M. Casillo · F. Colace · A. Lorusso · F. Marongiu · D. Santaniello (✉)
DIIn, University of Salerno, Fisciano, SA, Italy
e-mail: dsantaniello@unisa.it

M. Casillo
e-mail: mcasillo@unisa.it

F. Colace
e-mail: fcolace@unisa.it

A. Lorusso
e-mail: alorusso@unisa.it

F. Marongiu
e-mail: fmarongiu@unisa.it

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_8

1 Introduction

Italy has a unique Cultural Heritage (CH). Due to deterioration, this heritage must be preserved by preventing a severe lost that often cannot be recovered. Among its legislative principles, Italy states that cultural and environmental assets constitute Italy's cultural heritage, also defined as historical and artistic heritage. Cultural heritage is therefore identified as a national asset. This heritage must be protected and managed through protection, conservation and valorization. These actions are distinctly well-identified, but in practice, have levels of interconnection, as if they became a single guiding principle to be considered at all times in the context of any operation implemented on Cultural Heritage [1].

Protection is any activity aimed at recognizing, protecting and preserving an asset of our cultural heritage so that it can be offered to the collective knowledge and fruition. This phase is therefore expressed in recognition through verification or declaration of cultural interest of an asset, depending on its nature of ownership, protection and conservation. Conservation is any activity performed to maintain a cultural asset's integrity, identity, and functional efficiency in a consistent planned and coordinated way. It is therefore expressed in the study, intended as in-depth knowledge of the cultural heritage. Prevention, intended as a limitation of risk situations related to the cultural heritage in its context. Maintenance, considered as an intervention aimed at controlling the conditions of the cultural heritage to maintain it over time. Restoration, considered as a direct intervention on a cultural asset to recover its material integrity. Enhancement is any activity aimed at improving the conditions of knowledge and conservation of cultural heritage and increasing public access to it to transmit the values of which this heritage is the bearer [2–4].

The protection and preservation of CH is a particular concern as every work of art suffers some degradation with time. This deterioration may depend on several factors, such as external weather conditions, type of material, and human factors [5, 6]. One of the most widely used techniques for CH conservation is monitoring [7]. Monitoring allows intervention by implementing the necessary interventions over time, sometimes highlighting factors that can predict the problems related to the cultural property. The protection and conservation of each artefact present different criticalities due to a wide range of variable factors. These factors may include the climatic and microclimatic conditions to which the asset is exposed and the materials' physical, chemical, and mechanical characteristics. In this scenario, it seems impossible to intervene punctually, given the heterogeneity of the factors to be preserved to make Cultural Heritage conservation possible [8].

The advent of modern technology has provided a significant advantage in several aspects related to monitoring. In particular, the spread of low-cost technology has led to the possibility of having various devices that can communicate and interact with each other and humans. This phenomenon is known as the Internet of Things (IoT) paradigm [9, 10], therefore, many efforts in the literature have been made to protect the security of this paradigm and the protection of exchanged data [11–15]. This paradigm allows mapping reality by defining a virtual environment consistent

with the heterogeneity of the domain of interest due to the possibility of managing countless interconnected devices capable of generating digital copies or Digital Twin of the tangible elements [16–20].

Michael Grieves initially introduced the concept of Digital Twin, according to which the Digital Twin corresponds to the digital replica of any physical product [21]. A Digital Twin is the virtual representation of a physical entity, person or system, even a complex one. The digital component is connected somehow with the physical part. It can exchange data and information in a synchronous way (in real-time) or in an asynchronous way (in later times). The digital twin can evolve to become an authentic digital replica of potential and actual physical resources (physical twin), processes, people, places, infrastructure, systems and devices that can be used for multiple purposes. In general terms, the main characteristics of the digital twin are [21]:

The set of data and information in any way referable to the entities represented by the digital twin;

The connection between the elements of the physical component with the corresponding virtual part;

The possibility of ubiquitous access to data and information resources through the web, with the possibility of information search and analysis (big data, machine learning, artificial intelligence);

The exchange of data and information between the virtual and physical components, using sensors and actuators.

Digital twins integrate perfectly with IoT, artificial intelligence, and machine learning. Such models are able to create digital simulation models that update and change when their physical counterparts change. A digital twin can continuously learn and update from multiple sources. It can also represent, in real-time, the state, working condition, or location of objects or the physical system [22]. In fact, through IoT, it is possible to exploit several heterogeneous sensors that could contribute to the creation of Digital Twin to support the monitoring of Cultural Heritage [23]. Such sensors could define the mechanisms of degradation of an asset by acquiring information on:

Material alterations;
Structural alterations;
Environmental and micro-environmental conditions of the site.

Through the acquired data, the Digital Twin is able to exploit machine learning and artificial intelligence systems, to process the data and produce new knowledge or predict operational scenarios of CH conservation or predictive maintenance. In such a scenario, it would be interesting to introduce an architecture that is able to exploit the IoT paradigm to support expert users in monitoring, management and maintenance, which can ensure the survival of Cultural Heritage.

The present article is organized as follows. The next section deals with presenting some significant works that contribute to the presented field of approach. The third section deals with proposing the proposed IoT architecture. The fourth section deals

with validating the proposed model. The last section deals with conclusions and future developments.

2 Related Works

Many scientific contributions have been concerned with Cultural Heritage preservation through IoT-based systems [24, 25].

In particular, using a low-power network, an IoT-based architecture has been proposed to monitor large-scale monumental sites [26]. The first case study was applied to the medieval walls of the city of Siena. Also, in [27], the authors present a wireless architecture based on the IoT paradigm to monitor and control real-time lighting systems of the Scrovegni Chapel, Padua, Italy. Such a system is able to measure indoor lighting variations and environmental parameters, which are fundamental for the conservation of cultural heritage. These parameters are used to study correlations between artificial light, natural light and lighting performance, evaluating conservation and correlating this data with other critical environmental conditions.

As can be expected, the need to monitor condition information using sensors can be critical in assessing the declining state of CH. In addition, data analysis can facilitate preventive analysis approaches based on data mining, modelling, and machine learning techniques capable of analyzing the current state and predicting future events [28].

Furthermore, in [29], the authors present a three-tier IoT based architecture to monitor the Church of Archangels Michael and Gabriel in Sarajevo. The proposed architecture is capable of leveraging artificial intelligence algorithms such as Decision Tree, Support Vector Machine (SVM) to automate decisions for asset preservation.

3 The Proposed Approach

This work introduces a system framework for monitoring cultural heritage that can support expert users in the monitoring, management and maintenance phases. These phases aim at supporting expert users through suggestions and automatic decisions. In concurring with the stated goal, in Fig. 1, a multilevel architecture is presented that is able to exploit inferential engines that are capable of recommending the right action at the right time [30, 31]. These methodologies have already been used in the management of several complex scenarios [32]. They are an integral part of the framework in order to support expert users in the management concerning environmental conditions and available time. The proposed framework is composed of several layers shown in the following.

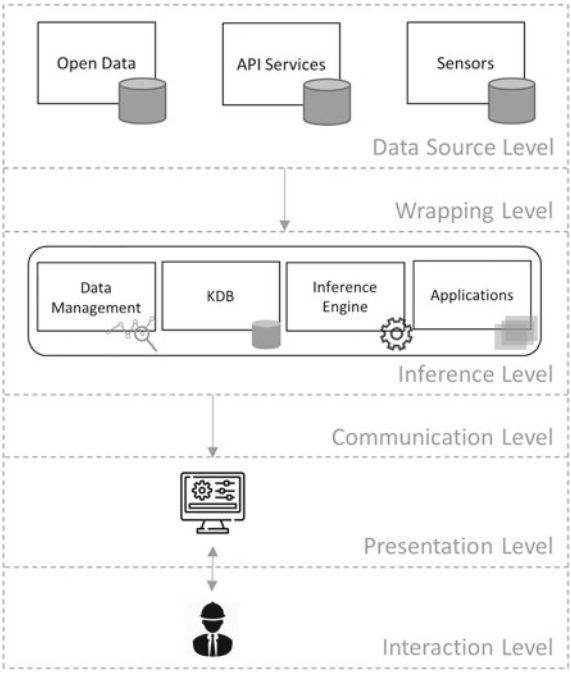


Fig. 1 The proposed architecture

4 Data Source Level

The Data Source Level ideally collects all the data sources functional to the monitoring, managing, and conserving cultural heritage. This level includes data from IoT-based sensors present in archaeological sites that can collect information about environmental conditions such as temperature, humidity, weather conditions in general and various types of images. Other data, valuable to the system, can come from open data services or external APIs. Such information can enrich the knowledge of expert users in terms of conservation and interventions to be carried out.

5 Wrapping Level

The Wrapping level deals with data transfer. At this level, data retrieval methodologies and protocols work to bring together information from different sources in order to transfer as much data as efficiently as possible. The collected data feeds the successive levels that will deal with the management and use.

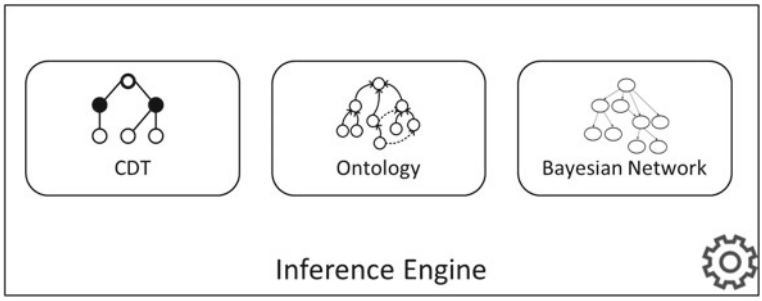


Fig. 2 Inference engine module

6 Inference Level

The inference level is developed through several stages, which can be schematized into different blocks:

Data management block.
Data storage block.
Data Inference Block.
Services Block.

These blocks are fundamental to the use of the available data. In particular, the data are pre-processed, making them homogeneous and available for storage in the knowledge database. The knowledge database stores all the data needed to feed the inference block. Within the inference block, data are processed to extract helpful information for preserving and managing cultural heritage. Within that block, several graph structures concur, as evidenced by Fig. 2. In particular, the Ontologies, able to model the domain of interest. The Context Dimension Tree (CDT), a tree capable of representing all possible contexts.

Moreover, the Bayesian Networks, a probabilistic graph approach able to predict the occurrence of certain events. In the last block, the applications block, several sub-modules manage the different applications of the system. The applications are specific for expert users, i.e., they can provide information and suggest autonomous decisions to support the preservation of cultural heritage.

7 Communication Level

This level deals with the communication of data to the next level. In particular, data are collected and conveyed through internal API services to feed the next level that deals with the presentation.

8 Presentation Level

The presentation level deals with the presentation of the different applications to the users. At this level, the system knowledge is represented through dashboards representing the Digital Twin of the Cultural Heritage. This tool represents the data acquired for monitoring and management and is able to suggest possible interventions for the safeguard of the cultural heritage in question.

9 Interaction Level

At this level, expert users can interact. In particular, expert users can act through the dashboards by performing actions or changing or supporting the autonomous choices of the system. The system can also collect information from users that is stored and reused within the inference engine [33, 34].

10 Experimental Results

In order to test the proposed framework, an experimental phase was conducted to test two fundamental aspects of the proposed methodology. In particular, the ability of the system to suggest and support expert users in interventions for the preservation of cultural heritage and the ability of the system to present the acquired information were tested. In order to achieve this goal, a prototype was developed with a server component and client components reachable through web applications. Technologies based on Python and Django REST Framework for the server part and Vue .js for the client part have been employed. The case study involved two rooms of "Casa Sannitica", a house of Roman times, which was hidden during the eruption of Vesuvius in 79 and found during the archaeological excavations of ancient Herculaneum. In the building different data were collected about temperature, humidity, moisture of architectural elements and images. The building was monitored through several sensors connected to a Raspberry Pi4 board to which were connected actuators to allow actions such as dehumidification and ventilation of the rooms. Communications to the central server were made through MQTT and HTTP protocols.

The experimental phase, therefore, took place in two steps. The first step was fundamental to understand the reliability of the system in supporting expert users. The data collected concerns approximately one year of observation between 2019 and 2020 was used to develop this experimental phase. The data collected covers the environmental and meteorological conditions (temperature, humidity, wall moisture, images etc.) and the actions taken to improve these conditions (ventilation, dehumidification, etc.). Over about one year, more than 4000 instances were collected. This dataset has been divided into a training set (75%) and a test set (25%). Using

the training set, through machine learning algorithms such as K2 Hill Climbing [35], it was possible to learn the Bayesian network structure then tested. The test took place on the dehumidification phase, thus understanding if the learned structure can automatically identify the need to prevent the dew point. The results, shown in Table 1, show that the system shows an Overall Accuracy that exceeds 88%, with Precision values of 70% and Recall of 72%. These results, although in a preliminary phase, are promising.

The second step of the experimental phase concerned the ability of the system to display the acquired information. To this end, 10 expert users were involved and were shown the web-based prototype of the system. The dashboard contains all the information that the system can provide concerning the asset to be monitored and the possibility to interact with it through different controls with remote procedure calls (Fig. 3). In order to test the system, after the interaction, a questionnaire was submitted to the users regarding several sections:

Presentation.
Reliability.
Performance.
Usability.

Table 1 Experimental results: confusion matrix

	Reference		
Prediction		Yes	No
	Yes	581	249
	No	224	2986
	Overall accuracy: 88.29%		

Fig. 3 System dashboard

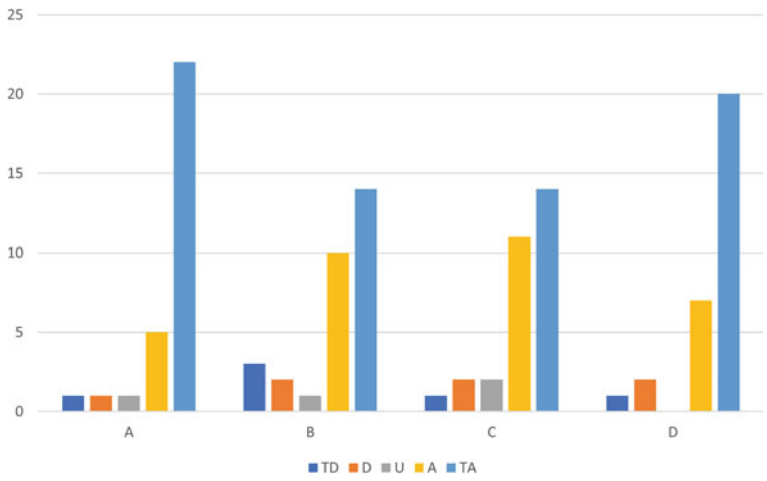


Fig. 4 Experimental results: statements answer

In addition to the reliability and performance of the system, the ability to present products and usability were tested, which provided us with important feedback on the system's ability to interact.

The questionnaire is based on the Likert scale and has five possible responses: totally disagree—TD, disagree—D, undecided—U, agree—A, totally agree—TA. The responses are shown in Fig. 4.

Although at a preliminary stage, the developed system showed promising user satisfaction ratings. In particular, significant results were achieved in Sects. 1 and 5, which represent, respectively, the ability to provide the correct information to users and the ability to interact. The performance of the system is still to be improved.

11 Conclusions

The objective of this work was to introduce a framework able to support expert users in the monitoring, management and maintenance of cultural heritage. This methodology included an experimental phase that involved the development of a prototype and the ability of the system to learn based on the data collected. The experimental phases, although preliminary, have collected promising results, and experienced users have expressed appreciation for the system's ability to monitor and support their activities. Future developments include an expansion of the database, which could improve the system and the increase of contextual parameters [36], which could improve the system's performance in terms of reliability. In addition, it is possible, based on the work presented, to be able to develop an application for the enhancement of cultural heritage for tourists and users. Such an application could include Digital Storytelling techniques [37] that can bring users closer, even remotely, to cultural

heritage. However, the reliability of the data used must be verified through using fact-checking techniques [38, 39].

References

1. Cano, E., Lafuente, D., Bastidas, D.M.: Use of EIS for the evaluation of the protective properties of coatings for metallic cultural heritage: a review. J. Solid State Electrochem. **14**(3). https://doi.org/10.1007/s10008-009-0902-6 (2010)
2. Bairagi, S., Custodio, M.C., Durand-Morat, A., Demont, M.: Preserving cultural heritage through the valorization of Cordillera heirloom rice in the Philippines. Agric. Human Values **38**(1). https://doi.org/10.1007/s10460-020-10159-w (2021)
3. Ghahramani, L., McArdle, K., Fatoric, S.: Minority community resilience and cultural heritage preservation: a case study of the Gullah Geechee community. Sustainability (Switzerland) **12**(6). https://doi.org/10.3390/su12062266 (2020)
4. Wang, Y.P., Dai, X., Jung, J.J., Choi, C.: Performance analysis of smart cultural heritage protection oriented wireless networks. Futur. Gener. Comput. Syst. **81**,(2018). https://doi.org/10.1016/j.future.2017.04.007
5. Reeder-Myers, L.A.: Cultural heritage at risk in the twenty-first century: a vulnerability assessment of coastal archaeological sites in the United States. J. Island Coast. Archaeol. **10**(3). https://doi.org/10.1080/15564894.2015.1008074 (2015)
6. Ubertini, F., Comanducci, G., Cavalagli, N., Laura Pisello, A., Luigi Materazzi, A., Cotana, F.: Environmental effects on natural frequencies of the San Pietro bell tower in Perugia, Italy, and their removal for structural performance assessment. Mech. Syst. Signal Process. **82**,(2017). https://doi.org/10.1016/j.ymssp.2016.05.025
7. Lucchi, E.: Review of preventive conservation in museum buildings. J. Cult. Herit. **29**. https://doi.org/10.1016/j.culher.2017.09.003 (2018)
8. Elfadaly, A., Attia, W., Qelichi, M.M., Murgante, B., Lasaponara, R.: Management of cultural heritage sites using remote sensing indices and spatial analysis techniques. Surv. Geophys. **39**(6). https://doi.org/10.1007/s10712-018-9489-8 (2018)
9. AlZu'bi, S., Hawashin, B., Mujahed, M., Jararweh, Y., Gupta, B.B.: An efficient employment of internet of multimedia things in smart and future agriculture. Multimedia Tools Appl. **78**(20). https://doi.org/10.1007/s11042-019-7367-0 (2019)
10. Ashton, K.: That 'internet of things' thing. RFID J. **22**(7), 97–114 (2009)
11. Gad, R., Talha, M., El-Latif, A.A., Zorkany, M., El-Sayed, A., El-Fishawy, N., Muhammad, G.: Iris recognition using multi-algorithmic approaches for Cognitive Internet of things (CIoT) framework. Future Gener. Comput. Syst. **89**, 178–191.https://doi.org/10.1016/j.future.2018.06.020 (2018)
12. Jararweh, Y., Alsmirat, M., Al-Ayyoub, M., Benkhelifa, E., Darabseh, A., Gupta, B., Doulat, A.: Software-defined system support for enabling ubiquitous mobile edge computing. Comput. J. **60**(10). https://doi.org/10.1093/comjnl/bxx019 (2017)
13. Lv, X., Hou, H., You, X., Zhang, X., Han, J.: Distant supervised relation extraction via DiSAN-2CNN on a feature level. Int. J. Seman. Web Inf. Syst. **16**(2). https://doi.org/10.4018/IJSWIS.2020040101 (2020)
14. Stergiou, C.L., Psannis, K.E., Gupta, B.B.: IoT-based big data secure management in the fog over a 6G wireless network. IEEE Internet Things J. **8**(7). https://doi.org/10.1109/JIOT.2020.3033131 (2021)
15. Zhang, W.-Z., Elgendy, I.A., Hammad, M., Iliyasu, A.M., Du, X., Guizani, M., El-Latif, A.A.A.: Secure and optimized load balancing for multitier IoT and edge-cloud computing systems. IEEE Internet Things J. **8**(10), 8119–8132 (2021). https://doi.org/10.1109/JIOT.2020.3042433
16. Abbas, K., Tawalbeh, L.A., Rafiq, A., Muthanna, A., Elgendy, I.A., Abd El-Latif, A.A.: Convergence of Blockchain and IoT for secure transportation systems in smart cities. Secur. Commun. Netw. **2021**, 1–13 (2021). https://doi.org/10.1155/2021/5597679

17. Al-Qerem, A., Alauthman, M., Almomani, A., Gupta, B.B.: IoT transaction processing through cooperative concurrency control on fog–cloud computing environment. Soft Comput. **24**(8). https://doi.org/10.1007/s00500-019-04220-y (2020)
18. Fuller, A., Fan, Z., Day, C., Barlow, C.: Digital twin: enabling technologies, challenges and open research. IEEE Access **8**,(2020). https://doi.org/10.1109/ACCESS.2020.2998358
19. Gupta, B.B., Quamara, M.: An overview of Internet of Things (IoT): architectural aspects, challenges, and protocols. Concurr. Comput. Pract. Exp. **32**(21). https://doi.org/10.1002/cpe.4946 (2020)
20. Tewari, A., Gupta, B.B.: Secure timestamp-based mutual authentication protocol for IoT devices using RFID tags. Int. J. Semant. Web Inf. Syst. **16**(3). https://doi.org/10.4018/IJSWIS.2020070102 (2020)
21. Grieves, M., Vickers, J.: Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems. In: Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches. https://doi.org/10.1007/978-3-319-38756-7_4 (2016)
22. Schroeder, G.N., Steinmetz, C., Pereira, C.E., Espindola, D.B.: Digital twin data modeling with AutomationML and a communication methodology for data exchange. IFAC-PapersOnLine **49**(30). https://doi.org/10.1016/j.ifacol.2016.11.115 (2016)
23. Jouan, P., Hallot, P.: Digital twin: a HBIM-based methodology to support preventive conservation of historic assets through heritage significance awareness. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences—ISPRS Archives, vol. 42(2/W15). https://doi.org/10.5194/isprs-archives-XLII-2-W15-609-2019 (2019)
24. Sornalatha, K., Kavitha, V.R.: IoT based smart museum using bluetooth low energy. In: Proceedings of the 3rd IEEE International Conference on Advances in Electrical and Electronics, Information, Communication and Bio-Informatics, AEEICB 2017. https://doi.org/10.1109/AEEICB.2017.7972368 (2017)
25. Valentini, F., Calcaterra, A., Antonaroli, S., Talamo, M.: Smart portable devices suitable for cultural heritage: a review. Sensors (Switzerland) **18**(8). https://doi.org/10.3390/s18082434 (2018)
26. Addabbo, T., Fort, A., Mugnaini, M., Panzardi, E., Pozzebon, A., Vignoli, V.: A city-scale IoT architecture for monumental structures monitoring. Measur. J. Int. Measur. Confed. **131**. https://doi.org/10.1016/j.measurement.2018.08.058 (2019)
27. Pierleoni, P., Belli, A., Palma, L., Valenti, S., Raggiunto, S., Incipini, L., Ceregioli, P.: The Scrovegni chapel moves into the future: An innovative internet of things solution brings new light to Giotto's masterpiece. IEEE Sens. J. **18**(18). https://doi.org/10.1109/JSEN.2018.2858543 (2018)
28. Narendra Kumar Rao, B., Bhaskar Kumar Rao, B., Challa, N.P.: Predictive maintenance for monitoring heritage buildings and digitization of structural information. Int. J. Innov. Technol. Explor. Eng. **8**(8) (2019)
29. Maksimovic, M., Cosovic, M.: Preservation of cultural heritage sites using IoT. In: 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH). https://doi.org/10.1109/INFOTEH.2019.8717658 (2019)
30. Colace, F., Lombardi, M., Pascale, F., Santaniello, D.: A multi-level approach for forecasting critical events in smart cities. In: Proceedings—DMSVIVA 2018: 24th International DMS Conference on Visualization and Visual Languages. https://doi.org/10.18293/DMSVIVA2018-002 (2018)
31. Fiorini, R.A.: Computational intelligence from autonomous system to super-smart society and beyond. Int. J. Softw. Sci. Comput. Intell. **12**(3). https://doi.org/10.4018/ijssci.2020070101 (2020)
32. Colace, F., Lombardi, M., Pascale, F., Santaniello, D.: A multilevel graph representation for big data interpretation in real scenarios. In: Proceedings—2018 3rd International Conference on System Reliability and Safety, ICSRS 2018, pp. 40–47. https://doi.org/10.1109/ICSRS.2018.8688834 (2019b)
33. Colace, F., De Santo, M., Lombardi, M., Santaniello, D.: CHARS: a cultural heritage adaptive recommender system. In: Proceedings of the 1st ACM International Workshop on Technology

Enablers and Innovative Applications for Smart Cities and Communities—TESCA'19, pp. 58–61. https://doi.org/10.1145/3364544.3364830 (2019a)
34. Cvitić, I., Peraković, D., Periša, M., Gupta, B.: Ensemble machine learning approach for classification of IoT devices in smart home. Int. J. Mach. Learn. Cybern. (2021). https://doi.org/10.1007/s13042-020-01241-0
35. Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Mach. Learn. **9**(4), 309–347 (1992). https://doi.org/10.1023/A:1022649401552
36. Colace, F., De Santo, M., Lombardi, M., Pascale, F., Santaniello, D., Tucker, A.: A multilevel graph approach for predicting bicycle usage in London area. In: Yang, X.-S., Sherratt, S., Dey, N., Joshi, A. (eds.) Fourth International Congress on Information and Communication Technology. Advances in Intelligent Systems and Computing, vol. 1027, pp. 353–362. Springer, Singapore. https://doi.org/10.1007/978-981-32-9343-4_28 (2020b)
37. Colace, F., De Santo, M., Lombardi, M., Mosca, R., Santaniello, D.: A multilayer approach for recommending contextual learning paths. J. Internet Serv. Inf. Secur. **2**(May), 91–102. https://doi.org/10.22667/JISIS.2020.05.31.091 (2020a)
38. Gupta, B.B., Tewari, A., Cvitić, I., Peraković, D., Chang, X.: Artificial intelligence empowered emails classifier for Internet of Things based systems in industry 4.0. Wirel. Netw. https://doi.org/10.1007/s11276-021-02619-w (2021)
39. Sahoo, S.R., Gupta, B.B.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Appl. Soft Comput. **100** (2021). https://doi.org/10.1016/j.asoc.2020.106983

Commercializing Blockchain in a Smart City: Autonomous Vehicles, Cryptocurrency, and Contract Law

Nidhee Kamble

Abstract This paper aims to provide a direction to the manifestation of a smart city, with blockchain-based implementations for economic, legal, and vehicular domains. The paper sees the three domains as three legs of the city, relevant to individuals and corporations, personal usages and businesses alike. Properties of blockchain and the consequences of its potential applications are kept as the focus of the paper. Since all systems in a city have to be commercialized to be viable for large-scale use, this paper presumes that innovations for such systems are required to be commercialized. This consideration shapes the approach with which such innovations are discussed, since scalability, security, and law-abidingness become essential. By analyzing the suitability of these technologies in their widespread adoption and their commercialization, this work intends to present direction for overcoming challenges in existing work in blockchain in economic, legal, and vehicular sectors.

Keywords Smart city · IoT · IoT security · Distributed ledger technology (DLT) · Autonomous vehicles (AV) · Intelligent systems · Cryptocurrency · Digital economy · Smart contracts

1 Introduction

The realization of the vision of a smart city presents challenges—and scope for innovation—in the fields of computer science, mathematics, mechanical systems and electronics, among others [1]. Included in this vision is the ability of such a city to optimize its resources for energy, transport, life supplies etc. through decision-making algorithms. This optimization also includes the preventive measures that shall be implemented for any security breaches, to allow continued provision of the aforementioned resources. In essence, a smart city can be seen as a concept where all amenities in the city are constructed such that they aid or replace the human effort

N. Kamble (✉)
VJTI, Mumbai, India
e-mail: ndkamble_b17@ce.vjti.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_9

currently employed to run those amenities, either to produce more efficient results or increase the result quality or both.

1.1 Contributions of This Work

This paper investigates three essential domains of a smart city, taking existing implementations and principles as bases: the economic and legal domains, and automation in the transportation industry, with the third domain seen as one that links the first two for the purpose of this paper.

The main contribution of this paper is its deliberate focus on the above three domains from a blockchain perspective for a smart city, in contrast with other works that have focused on these domains individually, or elucidated on the overall concept of a smart city and/or how blockchain will prove useful in it. More importantly, commercialization of any technology has specific challenges that are different in nature and magnitude from those discussed from a research or academic perspective—and this is what leads to this work being structured the way it is and discussing the topics and subtopics that it discusses. While academic perspectives may not always take into account the dissonance between research for its sake and practical, large-scale applications, this paper lays sole emphasis on *commercializing* blockchain for a smart city, hence taking academic developments and background to discuss large-scale, commercial use, which spans beyond the disciplines of science alone and ventures into financial, legal, and personal realms too.

Instead of seeing how newer implementations in these domains can be made to fit into the advancements in blockchain, this paper turns this perspective around to instead see how blockchain as a technology can evolve to fit the needs of economic, legal, and automotive developments in this sector. By discussing the inherent features of blockchain and what it can potentially offer, the paper establishes a sound base to enumerate and suggest measures to resolve the challenges that could arise in commercializing blockchain in the aforementioned domains.

1.2 Motivation

The motivation for selecting these three domains is their relevance to everyday routines of common people, corporations, or academia:

1. Legality: Every group of people, no matter their jobs, is governed by a set of rules;
2. Economic: Every such group gives and takes some resources of the city, generating value for the city's residents; this translates into circulation of fiscal resources as well;

3. Transportation Industry: Every such group uses certain media of transport—for personal or official usage or goods transport; the transportation media in turn run on laws specific to their industry—from manufacturing to procurement and usage, extending to insurance.

1.3 Paper Structure

The rest of the paper is organized as follows: Sect. 2 provides an overview of previous work along with the definitions for blockchain, autonomous vehicles, cryptocurrency, and smart contracts, so as to set the scope for this paper and establishing a base for Sect. 3. The properties of (different types of) blockchains and essential functionalities of autonomous vehicles are also discussed in Sect. 2 under the appropriate subsections. Section 3, Analysis, presents in its beginning the perspective that is used to remark upon the nature of and challenges in extant work. The three domains of the conceptual smart city are analyzed, in particular, through the lens of security, scalability, and legality. Section 4 categorically talks about the previous research in vehicular, economic, and legal domains, consolidating the key takeaways and directions for future work in the last subsection. Conclusion is presented in Sect. 5 with references at the end.

2 Background

Blockchain, a technology that has gained traction in the recent decade, has been used to address the security drawbacks or enhance related functionalities in the three domains aforementioned in this paper. Much has been done in that regard, with blockchain still adapting to suit applications that find widespread commercial use. To analyze how blockchain can be utilized in legal, economic, and vehicular domains in a smart city, previous related work must first be understood. To quantify the type of research work done in the primary domains addressed by this paper as enlisted in Table 1 in the past one year, i.e., 2020–2021, an advanced search of the Springer,

Table 1 Search results about relevant keywords integrated from Springer, ACM, and IEEE, in decreasing order of published work for each set of keywords

Keywords used	Springer	ACM	IEEE	Total
Distributed ledger	54	27,251	1217	28,522
Autonomous vehicles	397	5409	8158	13,964
Smart city	389	8052	1522	9963
Contract law	92	6237	91	6420
Blockchain legal	17	2821	137	2975

ACM Digital, and IEEE Xplore libraries was carried out, across French, Italian, Spanish, and German in addition to English.

Since the domains searched for are relatively new and fast-evolving, there is a marked difference in the work done on them in research from 2–3 years earlier (or older) as compared to the last one year. With time, the extent of implementation of previous concepts has increased, and ideation has been embellished, improving upon theories or suggestions stated in the previous years of research. All new work in the said domains—represented as keywords in Table 1—is, therefore, itself representative of previous years of research. It follows that to evaluate the current status of research on these domains, statistics of the last year are more relevant.

The scope of the search was: books, journals, and textbooks for Springer; journals, magazines, conferences, books, and early access articles for IEEE Xplore, and the entirety of ACM Digital Library. Searches were carried out twice for each of the keyword fields: once with the keywords specified in quotes and once without quotes.

Fields in Table 1 are arranged in decreasing order of the number of research documentation available. The term 'Contract Law', when searched for without quotes, yielded results pertaining to traditional contract law with and without blockchain applications—both of which are relevant to this paper, since it analyses the utility of blockchain-based solutions in the legal domain. The number of research documents indicated for Autonomous Vehicles in Table 1 include blockchain-related implementations, along with the domains of Internet of Things (IoT), cybersecurity, artificial intelligence, and proof of concept related papers; the purpose of including this term in the background section of this paper is to provide an idea about the extent of research on autonomous vehicles in comparison with other sections relevant to the scope of this paper.

The searches yielded some results that were noteworthy in the context of this paper: research analyzing blockchain for internet computing and for international security [2], papers discussing the social impacts of a smart city and usage of intelligent transport system with distributed ledger technology, and other papers on the economics of contract law, legally enforceable smart contract languages, and ethics and legal risks associated with autonomous vehicles.

2.1 Blockchain

Understanding blockchain as a technology and an ecosystem forms the base of all subsequent sections and sub-sections of this paper. The utility of blockchain-based applications in each of the domains in a smart city—economic, transport, and legal—is to be seen with respect to the properties of blockchain and the extent to which they are suited for these applications. Blockchains are capable of aiding and improving existing functionalities in industries, as elucidated upon later in this paper, and bringing about an infrastructure inversion—where new applications exist because of it, rather than it taking support from existing applications to thrive. Both of these implementations of blockchains shall make it burgeon in the coming years.

Definition. A blockchain [3] is an append-only list where all information is stored in blocks linked using cryptography. Subsequent blocks contain a cryptographic hash of their predecessor(s). This forms a link between the entire chain of blocks, where each block is representative of the state of data stored in all blocks before it. These hashes indicate block metadata, such as the timestamp and block header. Whenever a new block is to be added, the block contents (which are sets of transactions) are broadcasted to multiple nodes which validate the new block according to the rules of the consensus mechanism of the blockchain. Status of the block—valid or invalid—is broadcasted by the validating blocks to multiple nodes recursively, which in turn validate it themselves. Valid blocks are appended to the chain.

The functions of blockchains in a commercial setup can be broadly classified as the following, based on different aspects of operation:

1. Technical: As a distributed ledger or a database, that is maintained between the parties that share it—in either a closed or an open group;
2. Business: As a network for exchanging value between peers;
3. Legal: A transaction validation mechanism without the need of an intercessor.

In order to gauge the use cases of blockchain to tackle issues related to security, the cybersecurity aspects of blockchains themselves must first be understood in the context of confidentiality and integrity.

Confidentiality. Confidentiality of the data stored on the blockchain shall determine the type of network access that the blockchain shall offer:

1. Permissioned: Controlled network access with only relevant nodes participating in the consensus. Provides a certain level of control over participation. By that definition, private blockchains are also permissioned blockchains.
2. Public: Anyone can join the network and participate in the consensus; anyone can make copies of data.

Commercial usage of blockchain requires control of data access and disclosure, where only relevant parties store transaction data and others do not. For that reason, blockchain or distributed ledger frameworks which facilitate such controlled sharing of transactions [4] might find more commercial usage than those based on purist open blockchains.

Integrity. Integrity of data can be ensured by preventing any changes to its original state until it reaches the intended recipient. In other words, data should be immutable from source to destination. It must be noted that the change of form of data, for example from binary to radio waves, does not count as mutation since the content remains the same; with the only changes being the changes in encoding that data. Since data when processed becomes information, mutation is any change to the information, or the meaning that the data intends to convey. Mutability is the susceptibility of data to alteration.

Immutability is hailed as one of the key features of blockchains and is also the one that finds widespread applications commercially, as a property of either distributed ledgers or blockchains. To evaluate where this immutability can be utilized, it is

first necessary to understand its origin, and what the properties of blockchain actually mean with respect to immutability; in particular, the distinction between the terms *tamper tolerance*, *tamper resistance*, and *tamper free*. Innately, blockchains are designed to be tamper resistant, meaning that they are resilient to any attempt to modify the data on the main chain. It must be noted that this does not imply that it is impossible to modify the already-published state of data. That is certainly possible theoretically, but is not computationally feasible because of the consensus mechanism, which makes the design rather *tamper-resistant* than *tamper-free*.

Any changes that are made to the state of the data (after that data has already been accepted by all the nodes in the network) have to keep up with the changes in the longest chain of blocks in order to persist, since in any conflict, the longest chain is accepted as the main chain. Hence, to modify any previously accepted block in the chain such that the changed state persists, all the blocks succeeding the altered block must also be tampered. In essence, failure to reflect changes in the older blocks of the longest chain results in those changes being discarded, since no record of them appears on the ledger copy is accepted as 'true' by the network.

2.2 Autonomous Vehicles

The terms 'self-driving vehicles' or 'autonomous vehicles' refer to vehicles that navigate without human intervention, by integrating hardware sensors and software algorithms. NHTSA Guidelines [5] classify autonomous vehicles of level 4 and above as 'fully autonomous'. Level 4 of autonomous vehicles is defined in accordance with these guidelines as follows:

1. Level 0: No Automation
   This level consists of completely manual driving.
2. Level 1: Driving Assistance
   The vehicle can assist with steering or accelerating/braking but not both simultaneously. A driver is required to drive the vehicle.
3. Level 2: Partial Automation
   At this level, steering and accelerating/braking can be performed simultaneously but the driver must monitor the driving environment and perform the remaining driving operations.
4. Level 3: Conditional Automation
   At this level, the car can perform all aspects of driving, but a driver must be present in case the system requests so.
5. Level 4: Full Self-Driving Automation
   By design, the vehicle requires destination input from the driver but does not expect the driver to supervise the functions crucial to safety, such as monitoring road conditions and performing driving functions, hence performing all of them itself. The vehicle alone is responsible for its own and the driver's safety.

'Intelligent Transport System (ITS)' is an infrastructure where vehicles are connected with each other using smart devices. Autonomous vehicles, as a part of or independent from ITS, have a plethora of applications in different sectors, from vehicle-sharing enterprises (like taxi services or carpooling) to fleet management systems.

For the purpose of this paper, we will consider autonomous vehicles to be of Level 3 and higher. Network infrastructure and service architecture for AVs can vary, using a multitude of technologies integrated with each other thus have various components. Although these components all have different uses, the services and infrastructure of an AV essentially give five major functionalities to it:

1. Vehicular state estimation (static/dynamic);
2. Information retrieval about the surrounding (static/moving objects);
3. Information collection on driver/occupant state (to prevent casualties or report them);
4. Communication with other vehicles and other infrastructure (traffic lights or stop signs);
5. Enabling access to a Positioning System (perhaps GPS).

2.3 Cryptocurrency

Digitizing the economy in a smart city can be taken a step further by using cryptocurrency. All currencies are tokens representing exchange of value in the ecosystem within which they are used. Cryptocurrency is cryptographic currency: it is a binary data token that represents value exchange, with the ownership records of each such token stored cryptographically on a computerized database to secure transaction records, regulate the creation of this currency, and verify the transfer of coin ownership. [6]

Altcoins. The cryptocurrencies and other cryptographic digital assets which are not bitcoin are collectively referred to as 'altcoins', short for 'alternative cryptocurrencies'.

Cryptotokens. Units of value exchange used in blockchains for functions other than payments, e.g. in decentralized applications or smart contracts, apart from the primary cryptocurrency of the respective blockchain are called cryptotokens. These are issued on top of the main blockchains, such as in smart contracts on Ethereum [7].

2.4 Smart Contracts

Currently existing smart contracts can be thought of as structured digital agreements. Although they are 'contracts', their implementation innately lacks the constructs necessary for their execution and enforcement.

Certain papers also establish certain naming conventions to refer to and contrast smart contract setups and proposed technology. One such naming convention is as follows:

1. Conventional contracts (CCs), that are not machine readable and challenging to interpret in a legal environment;
2. Self aware contracts (SACs), that are similar to CCs with respect to legal enforceability supported by blockchain and multi-agent systems. The storage is designed to be immutable and the information is managed by software agents. SACs are therefore, in principle, suited for a trustless environment. Proposing the use of SACs in an environment with some degree of trust naturally results in a gap not addressed by the principles behind SAC's operation. An SAC framework [8] can bridge the gap by allowing collaboration between multiagent systems in a peer-to-peer, 'blockchain-agnostic' economy.

Some research papers introduce the terms 'strong' and 'weak' contracts on the basis of costs of their modification. Strong smart contracts have discouragingly high costs of annulment and modification; weak smart contracts do not. With respect to traditional legal practices, if a court is able to easily modify a contract post its execution, papers define it as a weak smart contract, and if the large cost associated with altering a contract makes doing so infeasible or otiose for a court, then the contract is defined as a strong smart contract.

All commercial applications—digital or not—for public use at any scale in a smart city would, ultimately, be governed by rules. This paper proposes that the traditional contract law that governs these legislations should itself also leverage the features of blockchain for two reasons: one, to reduce the disparity between ideation and implementation in the digital and non-digital domain, so as to be better able to intervene when required; and two, for the features' own sake.

3 Analysis

This section opines on the security, scalability of blockchain-based extant work in the autonomous vehicle and cryptocurrency sectors, and separately comments upon the legal aspects of these works. This section also states the challenges that developments in these domain present.

In general, any analysis of the goals that underlie the digitization of the economy seeks to evaluate, in essence, if digitization is a phenomenon in evolution, an innovative toolkit, or a 'qualitatively new state of the macroeconomic system' [9]. The goals of economic activities in digital economies shall, like the traditional ones, be envisioned by and be the subject of humans, who still continue to take all essential decisions even in fields with the advent of technology replacing human activities.

The judging of the quality of—if not acceptance or rejection of—the course or the result of (an) action(s) taken by an algorithm is also done by humans. To specifically point out instances where complete autonomy given to machines has resulted

in disastrous outcomes to establish a generalization is technically flawed. Computer programs, although capable of aggregating factors that are orders of magnitude more than subjective human experiences over limited timespans, end up specializing only in the specific domain in which they were made to excel; near-human capabilities require a step further towards 'general' intelligence, something that current advancements have attempted to achieve but haven't yet fully achieved. Consequently, the decisions taken by humans in automated processes are pivotal and can extend beyond final and preliminary activities such as defining the start and endpoint of a journey in an autonomous vehicle.

3.1 Blockchain and IoT

Instead of directly applying blockchain and distributed ledger systems to IoT topologies, these decentralised data structures linked cryptographically must be reconstructed ground-up from its principles to let IoT fully leverage its features by providing operational details crucial to data collected and transmitted from IoT nodes: contents and validation of a transaction, mechanism of a node joining and publishing on the network, or significance or modification in consensus mechanism. Existing blockchains dominating technological advances at present are configurations that are consequences of specific requirements of the above parameters, and hence should not be taken as the only possible solutions.

To preserve integrity, embedded systems working with sensor data can hash the readings along with the metadata, like timestamp, to be sent to a gateway which sends these copies to multiple servers that store or process the data. The gateway can rehash received data to verify its link with the source before transmitting it to servers.

The features offered by blockchain to protect monetary transactions can also be availed by data recording and transmission—a use case more relevant in embedded systems for IoT. Blockchain can be used for not just distributed, tamper-resistant systems, but also for communication security between IoT nodes for autonomous interaction. Integrity of data transmitted between nodes at each stage can be validated in accordance with protocol established for the blockchain network. This data may be related to financial transactions, but can also be raw or processed data from sensor nodes. Nodes that are endpoints can make decisions about data validity themselves before transmission instead of relying on centrally-controlled mechanisms. These use cases would be suited in systems where different trust relationships exist (trust may be partially present or completely absent and an objective record of transactions (data or monetary) is required.

Examples of such applications might be in different sectors of autonomous vehicle industry, embedded systems for residential areas or in agriculture for soil analysis. By having a consistent, tamper-resistant record of this data, it can later be used for deriving patterns. Using blockchain in IoT necessitates high levels of coordination

in embedded systems in IoT networks between the chosen implementation and low-level security modules.

3.2 Security

Cyber risks pose a threat to both individuals and corporations. Anything digital is susceptible to cyber risks. With rise in cyber risks, financial losses due to these risks can—and in most cases do—rise. The aspects of businesses which could be affected by (the changes in state of) data should be analyzed with respect to their storage and transfer mechanisms, e.g., different storage areas for persistence, message calls, and computation in solidity [10]. A focus on privacy, by design, translates to not having unnecessary data [11].

Blockchains can offer protection from frauds, but are difficult to implement because of the overhead of multiple layers of security required to facilitate this. A further direction to research could be to find out if, despite this protection, integrity violation is possible using quantum computing.

Cyber Risks in Smart City. IoT devices, instrumental to several functionalities of autonomous vehicles, can become single points of failure—a shortcoming that can be addressed by using blockchain to distribute these points, having active sites even if one of them is compromised. In the situation where one of the distributed nodes fails, blockchain can serve as a means for tracing changes. Especially in a sharing economy based on IoT [12], although there are no single points of failure, the distribution of vulnerable nodes could make the framework more prone to attacks.

Confidentiality and accessibility can be regulated by blockchains by providing authentication services [13] for the IoT devices used in autonomous vehicles. Security during transactions—in the domain of autonomous vehicles and outside—is crucial to make the payment methods safe for widespread usage. Public releases of all systems where legal tenders are exchanged introduce a great risk of the value in circulation being lost lest there be any backdoors or attacks that cannot be recovered from without significant financial loss. The effect of such a catastrophe would be far-reaching—to the residents of the city, corporations, and possibly even international trade—to everyone who is a user of these systems.

Blockchain and Security. The strong privacy mandates of corporations using permissioned blockchains can further underscore the trilemma of 'decentralization, scalability, and security': The greater the decentralization and security, the more shall be the tradeoffs made for scalability. Greater security in blockchain contexts means more computationally intensive operations, requiring greater hardware resources. The software and hardware required to support increased levels of decentralization and security might make the architecture difficult to scale.

Confidential computing, where data can get processed by other nodes without losing privacy and control of that data, can be employed where third-party processing of data is required with the third-party getting only computation permits and no visibility or control over data that is processed. It can therefore be seen as a possible

solution to outsourcing the computations required for permissioned blockchain. It must be decided, however, how critical the data to be outsourced is, and what the consequences could be in the unlikely event of unauthorized access of that data while outsourcing.

3.3 Scalability of Blockchain Applications

The thermodynamic costs associated with Proof of Work in [14] seem too demanding to facilitate transaction validate at the swift rate that could be required by riders in the vehicle sharing system. Additionally, the resources necessary for calculating proof of work may impair the mobility of the vehicles in the system, or introduce a lag in the other computationally-intensive tasks. The simulation of the system done on Windows 10—an operating system that is not lightweight—cannot be considered an accurate measure of the implementation, if carried out. A simulation intended to give results that could closely match with the implementation of the proposed ITS could be done in an operating system having minimal or no overhead, using only the drivers necessary for hardware related to inter-vehicular communication. A good candidate for such an operating system could be any that is Arch Linux [15] based, allowing precise customization depending on requirements. The authors do mention the most apparent drawback of the concept: the responsible entity in case of occurrence of an accident.

Scalability becomes even more necessary when seen in context of mass usage. Two such instances and a large volume of cryptocurrency transactions, and public transport systems where autonomous vehicles are aided by blockchain. If cryptocurrency is to be (one of) legal tender(s) in a smart city, the blockchain model for the cryptocurrencies used should be equipped to handle the massive transaction volume without delays negatively affecting the purpose that the transactions are made for. These applications can include transactions related to paid parking, groceries and charging stations, to name a few. For the public transport system of autonomous vehicles, the blockchain consensus mechanism and the hardware should be able to handle responses from the vast number of IoT devices on the vehicles, in real time if required, and process it without loss in service or revenue.

3.4 Technology and Legality

Innovative technology may not necessitate innovative jurisprudence. Although traditional legal analysis can help craft simple rules as a framework for this complex phenomenon, the main problem in contract law is what happens when the outcomes of the smart contract diverge from the outcomes that the law demands.

Much of the literature on self-help in contract law has dealt with how a party that has been aggrieved can remedy the wrong that has been committed against it.

The development in laws for smart contracts has so far focused attention on the harms done to the breaching party, ensuring that party autonomy takes a backseat to other norms that are enforced by the society (which includes prevailing traditional legislature). Some researchers, therefore, examine the costs to breaching parties, potential benefits to non-breaching parties and society at large, and the limits that the state could place on the use of smart contracts.

The authors of [14] do mention the most apparent drawback of the concept of the proposed ITS: the responsible entity in case of occurrence of an accident. Further, the question of which jurisdiction an accident would fall under, if occurred, also remains—it is difficult to determine if that jurisdiction would be the one that governs developers of the mobility algorithms of the ITS, the one where the system was built, where the deployed system is hosted, or where the accident took place—if all of the jurisdictions are different and governed by different laws.

An interesting concept to consider is how the import and export relations between the smart city and other legislations could change once the smart city starts accepting cryptocurrencies as legal tender. Since the very concept of cryptocurrencies is governed by code and not jurisdictions, laws shall have to think of if there must be a balance between the foundational principles of cryptocurrency and their practicality. Especially in the case of open permissionless blockchains, like Bitcoin, governance becomes difficult, since it was thought to be unnecessary to the inception of that blockchain and its cryptocurrency. Cryptographic money certainly has its advantages, but previously ideated models may not be suited for everyday use by everyone. If the laws require a different technology, it must first be ensured that such technologies are feasible. These implementations should also consider how city-specific regulations on cryptocurrencies could affect arbitrage.

4 Aspects of a Smart City

4.1 Vehicular

Surveys enumerate the different use cases related to autonomous vehicles coupled with blockchain technology such as providing shared storage, enhancing security, optimizing vehicular functionalities and enhancing related industries [16], citing that these use cases can spawn new applications like searching for nearby charging stations and rented parking slots, autonomous taxis, and automated tolls, specifically for autonomous vehicles of level 3 or above, replacing traditional human-oriented jobs. The monetary token proposed for transactions in these applications is cryptocurrency. Oft-cited concerns for such transactions are scalability and feasibility of computation. In the case of cryptocurrencies native to smaller blockchains, it is not difficult to aggregate the computational power for these attacks, and such attacks are feasible. The proposals, therefore, advise exercising caution before selecting a cryptocurrency for payments in the autonomous vehicle sector.

An ITS based on blockchain [14] for vehicle sharing resolves the dependence on third parties for checking availability of sharing respective vehicles depending on the route and destination. In comparison with existing vehicle-sharing services, the approach proposed in the paper [14] is free from driver influence: instead of letting the vehicle drivers choose their passengers, the system gives AVs full authorization to select any customer. The transactions related to the trip are carried out using smart contracts, the consensus algorithm being Proof of Work.

4.2 Economic

The Bitcoin whitepaper gave impetus to conceptualization—and ultimately implementation—of new cryptocurrencies, the varieties of which have only increased since 2009: Cryptocurrencies have been launched and fueled as assets that wax and wane in different time frames in the market [17]. This claim can be evaluated in the light of developments in the cryptocurrency sector up till the time of this writing, where the term 'cryptocurrency' encompasses altcoins and cryptotokens.

With development in cryptocurrency allowing for money laundering and innovative utilization alike, it is considered difficult at present to envisage a panacea, a universal framework that governs all such transactions while preventing any unintended use. Moreover, this fluctuation in the fiat prices of cryptocurrencies is an obstacle to the realization of such a framework that would supplant current national and international fiscal practices. Although anti-money laundering methods have been oft-quoted and analyzed, a one-size-fits-all solution is not presently possible, especially given the lack of stability in the cryptocurrency domain. Some suggestions for achieving stabilization emphasize limiting cryptocurrency emission by time or volume [17]. With the state's regulation of the economy, the administration would play a pivotal role in risk management related to any legal tender; if cryptocurrencies should be governed by such sovereign rules, is an entirely different question altogether.

4.3 Legal

Alternative definitions of 'smart contracts' have also been proposed, which can be compared and interplayed across research papers. In some researches that define a template for smart contract creation, traditional means of enforcement are also defined, as those through institutions like arbitration or courts of law—which ultimately is human intervention. A significant obstacle to the use of smart contracts for legal practice is the lack of adequate regulations and the absence of a proper legal framework for the blockchains that implement them. For a smart contract to be enforceable as a legal contract, it has to meet several conditions imposed by the law, such as the capacities of multiple parties, their mutual assent, and their consideration

[18]. It may be difficult to provide mutual assent with a smart contract without having a Trusted Third Party.

Self-help [19] remedies have been defined as 'legally permissible conduct that individuals undertake to absent the compulsion of law and without the assistance of a government official in efforts to prevent or remedy a civil wrong.'

4.4 Key Takeaways and Future Direction

1. Blockchain can further be used to optimize many functionalities of autonomous vehicles [16]: paid parking using parking pools and non-fungible tokens for rental, demand car-rental [12], verification of insurance records of autonomous vehicles using a permissioned blockchain, and tracking of autonomous vehicle lifecycle, and in the supply chain to maintain and track vehicle inventory.
2. There is a lot of scope for insurance-based companies to utilize blockchains to make faster processing and payouts using smart contracts. Mainstream companies like Tata Consultancy Services, Deloitte, and IBM too have adopted blockchain for insurance. Insurances where payouts is in cryptocurrency or cryptotokens can be the next step. Further, different kinds of insurances, like risk prevention, reinsurance, and life insurance, can be modified to leverage demand insurance frameworks [12] to increase efficiency.
3. Smart contracts, at the time of this writing, lack suitable obligation and the right constructs for execution and enforcement, sans the dynamics of legal relations. One possible resolution for this issue could be establishing ontologies to define primary contractual components. Although such ontology exists for SACs [8], it can be extrapolated to hold true for all types of smart contracts in general.

This paper has seen the legal, economic, and autonomous vehicle domains as the three legs on which a conceptualized smart city would stand. There are several areas that lie outside the scope of this paper and its suggestions, like: changes in legislation that accommodate the entity responsible in different cases of accidents between an AV and a person, or the protocols that monetary transactions between individuals and corporations must adhere to so as to include cryptocurrency besides the state currency. Consequently, it is possible that novel ideas are not found to be suited for full-fledged public utility in a smart city and may hence be better used within their own ambits by a significantly smaller section of people. Nonetheless, it is hoped the perspectives provided by this paper (overview, challenges, and suggestions) shall incentivize new works in the areas mentioned in this paragraph as well. It is hoped that in that direction, a sound basis for the functioning of a truly 'smart' city can be established, leading way to newer applications facilitated by such infrastructure.

5 Conclusion

Blockchain implementations for embedded systems in a smart city need not be limited to existing frameworks like Ethereum and Hyperledger. Bespoke implementations that are tailored to suit the functional and performance needs of these systems can offer more flexibility and customisation. To facilitate such execution, a conceptual threshold must first be crossed: to see blockchain technology not from a purist perspective but as a development that can be remodelled according to use cases without a computationally intensive consensus mechanism.

In the economy of a smart city thus adumbrated, financial services are managed by automated cognitive algorithms that take 'trusted' decisions in which stakeholders operate. This consensus is studied by algorithms to be executed later without human involvement. Implementing such algorithms as smart contracts includes security features for inter-contract communication. For decisions made by such systems, near or far-range location information from IoT devices could be gathered using suitable protocols—like radio-frequency identification, near-field communication, or long-range—in the case of autonomous vehicles for example. It is hoped that with the insights on the features and applications of blockchain with respect to the mentioned sectors shall serve as a starting point for further research, extending beyond blockchain and IoT to legal ontology, digital economics, and confidential and distributed computing. A smart city can additionally take the aid of intelligent agents to enhance automation in existing domains and bring it to new ones.

The technological advancements in other aspects of a smart city shall, eventually, extend to the creation of aspects that may not be recognized by current contract laws and other legal practices. One possible addition to or change in traditional legal setups could be the use of smart contracts. Surveying what governs these advancements is necessary to facilitate the appropriate changes in the respective environments since smart contracts are not only administered by their programming logic but also by the state of law.

References

1. Hall, R.E., Bowerman, B., Braverman, J., Taylor, J., Todosow, H., Von Wimmersperg, U.Thu.: The vision of a smart city. United States. https://www.osti.gov/servlets/purl/773961
2. Vestergaard, C.: Blockchain for international security—the potential of distributed ledger technology for nonproliferation and export controls | Cindy Vestergaard | Springer. https://www.springer.com/gp/book/9783030862398. Accessed 26 Aug 2021 (2021)
3. Blockchain—Wikipedia: https://en.wikipedia.org/wiki/Blockchain. Accessed 09 Jan 2021
4. Kharitonova, E., Kharitonova, N., Pulyaeva, V., Shaidullina, V.: Revealing best practices in legal regulation in the sphere of blockchain technology. In: Proceedings of the 2nd International Scientific Conference on Innovations in Digital Economy: SPBPU IDE-2020 (SPBPU IDE '20). Association for Computing Machinery, New York, NY, USA, Article 1, 1–8. https://doi.org/10.1145/3444465.3444475 (2020)

5. "Dot/NHTSA policy Statement Concerning Automated Vehicles". http://www.nhtsa.gov/staticfiles/rulemaking/pdf/Autonomous-Vehicles-Policy-Update-2016.pdf. Accessed 14 Feb 2021 (2016)
6. Cryptocurrency—Wikipedia: https://en.wikipedia.org/wiki/Cryptocurrency. Accessed 1 Sept 2021
7. Ethereum—Ethereum Home: https://ethereum.org/en/. Accessed 1 Sept 2021
8. Norta, A.: Self-aware smart contracts with legal relevance, pp. 1–8. https://doi.org/10.1109/IJCNN.2018.8489235 (2018)
9. Astafyev, I.V., Sokolov, D.P.: Digital economy: beautiful, but imaginary, concept. In: Popkova, E., Sergi, B. (eds.) Digital Economy: Complexity and Variety vs. Rationality. ISC 2019. Lecture Notes in Networks and Systems, vol. 87. Springer, Cham. https://doi.org/10.1007/978-3-030-29586-8_27 (2020)
10. Introduction to smart contracts, solidity. Source: https://docs.soliditylang.org/en/v0.8.7/introduction-to-smart-contracts.html
11. Tambay, P.: Talk on blockchains and cybersecurity [Video]. https://www.youtube.com/watch?v=HD9IqpWIehI (2021)
12. Rahman, M.A., Rashid, M.M., Hossain, M.S., Hassanain, E., Alhamid, M.F., Guizani, M.: Blockchain and IoT-based cognitive edge framework for sharing economy services in a smart city. IEEE Access **7**, 18611–18621 (2019). https://doi.org/10.1109/ACCESS.2019.2896065
13. Hammi, M.T., Hammi, B., Bellot, P., Serhrouchni, A.: Bubbles of trust: a decentralized blockchain-based authentication system for IoT. Comput. Secur. **78**, 126–142. ISSN 0167-4048. https://doi.org/10.1016/j.cose.2018.06.004 (2018)
14. Abubaker, Z., Gurmani, M.U., Sultana, T., Azeem, M., Iftikhar, M., Javaid, N.: Decentralized mechanism for hiring the smart autonomous vehicles using blockchain (2019)
15. Arch Linux—Arch Linux: https://archlinux.org/. Accessed 1 Sept 2021
16. Kamble, N., Gala, R., Vijayaraghavan, R., Shukla, E., Patel, D.: Using blockchain in autonomous vehicles. In: Maleh, Y., Baddi, Y., Alazab, M., Tawalbeh, L., Romdhani, I. (eds.) Artificial Intelligence and Blockchain for Future Cybersecurity Applications. Studies in Big Data, vol. 90. Springer, Cham. https://doi.org/10.1007/978-3-030-74575-2_15 (2021)
17. Bystriakov, A.J., Guirinskiy, A.V., Nan, T.N., Hidar, S., Din, L.C.: Crypto currencies and possible risks. In: Popkova, E., Sergi, B. (eds.) Digital Economy: Complexity and Variety vs. Rationality. ISC 2019. Lecture Notes in Networks and Systems, vol. 87. Springer, Cham. https://doi.org/10.1007/978-3-030-29586-8_21 (2020)
18. O'Shields, R.: Smart Contracts: Legal Agreements for the Blockchain, 21 N.C. Banking Inst. 177. http://scholarship.law.unc.edu/ncbi/vol21/iss1/11 (2017)
19. Raskin, M.: The Law and Legality of Smart Contracts (September 22, 2016). 1 Georgetown Law Technology Review 304. Available at SSRN: https://ssrn.com/abstract=2959166 or https://doi.org/10.2139/ssrn.2842258 (2017)

Finding All Shortest Meaningful Meta-Paths Between Two Vertices of a Secured Large Heterogeneous Information Network Using Distributed Algorithm

Phuc Do

Abstract Discovering relationships between vertices in a secured information network is an important task in information network analysis. In HIN, meta-path, or a sequence of vertex types and edge types connecting two vertices. Path instance of a meta-path is path in HIN that satisfies the meta-path. The length of meta-path is the number of relations (edges) in this meta-path. Meaningful meta-path is a meta-path with at least one path instance. Recent works on meta-path discovery mainly focus on in-memory algorithms that fit in only one computer. In this chapter, we propose distributed algorithms to discover all shortest meaningful meta-paths between two vertices of a large HIN using Apache Spark. Shortest meaningful meta-path is a meaningful meta-path with shortest length. We employ a scalable implementation of the Distributed Breadth-First Search (D-BFS) algorithm as a baseline approach. Finding all possible shortest paths in a large HIN can be time consuming. Therefore, we propose a novel algorithm called shortest meaningful meta-path based search (S-MPS). S-MPS first searches all shortest meta-path candidates between vertices in the graph of the network schema of HIN. We conduct experiments on DBLP data set to prove the efficiency of our proposed S-MPS algorithm over D-BFS.

Keywords Apache spark · Distributed algorithms · D-BFS algorithm · S-MPS algorithm · Shortest meaningful meta-paths

1 Introduction

Heterogeneous Information Networks (HINs) is a graph model in which both vertices and edges are associated with types. Many real-world data can be represented as a HIN such as DBLP, FreeBase, YaGo. The DBLP bibliographic network, for instances, is a HIN containing publication information. Each vertex in DBLP has one of the

P. Do (✉)
University of Information Technology—Vietnam National University Ho Chi Minh City; Quarter 6, Linh Trung Ward, Thu Duc District, Ho Chi Minh City, Vietnam
e-mail: phucdo@uit.edu.vn

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_10

following vertex types such as (Paper), (Author), (Venue), (Topic) and (Word). Based on these vertex types we have several edge types. For example, one edge type is [Write] connecting two vertex types (Author) and (Paper). Another edge type is [PublishedAt] connecting (Paper) and (Venue). A graph consisting of vertex types and edge types is called a network schema. A network schema usually has significantly smaller number of vertices and edges than its corresponding HIN.

Due to retaining rich type information, analyzing relations between vertices in HINs is an important analysis task. In HINs, meta-path, or a sequence of vertex types and edge types connecting two vertices, is introduced to express relations between vertices [1]. The relation between vertices is considered as an edge between these vertices. We use the number of relations or edges in a meta-path to indicate the length of meta-path. For example, in DBLP meta-path (Author) $\xrightarrow{\text{Write}}$ (Paper) is a meta-path of length 1 which explains a direct relation between author and paper. An example of meta-path of length 2 of DBLP is (Author) $\xrightarrow{\text{Write}}$ (Paper) $\xrightarrow{\text{PublishedAt}}$ (Venue), which explains indirect relation between (Author) and (Venue). The longer a meta-path, the more complex relation it can express. In practice, short meta-paths are often considered since long meta-paths are shown to not be useful [1, 2]. Moreover, not all meta-paths have their corresponding path instances in the HIN in consideration. Such meta-paths are not meaningful since they do not express any actual relationship. Hence, in our work we only consider meaningful meta-path, or meta-path with at least one path instance. In these meaningful meta-paths we also consider the shortest meaningful meta-paths with the shortest length. From the shortest meaningful meta-path, we can infer the close relation between two vertices of HIN. We use shortest meaningful meta-path as a template to generate the question expressing the relation between two vertices and the answer of this question is the path instance of this meta-path [3].

Meta-path has been used extensively in many analytic applications in HIN such as similarity measure [4], clustering and classification [5–7], community detection [8], NLP question answering system [9], the semantics of HIN [10]. These applications rely on meta-paths specified in advance by domain experts, which may have limitations. First, relying on domain experts can be challenging in practice due to their scarce availability, or high labor cost. Secondly, the number of meta-paths between two vertices also grows accordingly to the size of HINs, making it more labor-intensive for manually selecting meta-paths on large HINs. Therefore, there have been works in literature focusing on algorithms to discover meta-paths automatically and efficiently [11–15]. These algorithms are designed with a common assumption that the input HIN can be loaded into a single machine. This assumption imposes an implicit constraint on the scale of the input HIN, which limits the applicability of these algorithms in real-world contexts of large HINs.

In contrast with the existing approaches of solely relying on the processing capacity of a single machine, we propose distributed algorithms [16, 17] to efficiently discover all the shortest meaningful meta-paths between two vertices in large HINs. An intuitive solution to discover shortest meaningful meta-path is to explore all shortest paths connecting the two given vertices, and subsequently generate meta-paths

from those shortest paths. From this intuition, we develop a distributed algorithm based on Breadth-First Search (D-BFS) using Apache Spark. D-BFS algorithm firstly searches for all possible shortest paths between two vertices. From the result of the returned shortest paths, D-BFS algorithm would then construct meaningful meta-paths accordingly. Even though Apache Spark is efficient in finding paths between vertices, finding all the shortest meta-paths is still time-consuming on large HINs. To avoid the bottleneck in finding shortest paths directly on the input HIN, we propose another novel algorithm called meaningful meta-path based search (S-MPS) algorithm. S-MPS algorithm is designed from the observation that meta-paths can be generated from a network schema of a HIN, which usually has significantly smaller size than the HIN itself. There are three steps of S-MPS algorithm. First, we generate a network schema from the given HIN, which is a lightweight task. Then, we explore all shortest meta-path candidates from this network schema. Finally, we leverage a distributed algorithm by using RDD of Apache Spark for generating path instances of a given meta-path candidate to verify whether this meta-path candidate is meaningful or not. S-MPS algorithm enjoys the benefit of generating meta-paths from network schema; hence it is more efficient than D-BFS. We conduct extensive experiments on a DBLP data set to prove the outperformance of S-MPS algorithm over D-BFS algorithm [18, 19]. Our contributions of this chapter are summarized as follows:

- We present a scalable implementation of D-BFS algorithm using Apache Spark, which directly operates on a given large HIN to find shortest meaningful meta-paths.
- We propose a novel distributed algorithm called S-MPS algorithm. This algorithm operates on both a given HIN and its network schema to find shortest meaningful meta-paths. By leveraging the network schema, S-MPS algorithm can achieve more efficiency than D-BFS algorithm in meta-path discovery.
- We conduct experiments to demonstrate the efficiency of S-MPS against D-BFS on a large DBLP dataset.

The rest of this chapter is organized as follows. Section 2 discusses related work to ours. In Sect. 3, we introduce fundamental definitions in our work. In Sect. 4, we present the Distributed Computing with Apache Spark. In Sect. 5 and 6, we describe details of developing the D-BFS algorithm and the S-MPS algorithm on Apache Spark. In Sect. 7, we evaluate and compare the efficiency of D-BFS algorithm with S-MPS algorithm. We report and discuss our experimental results in Sect. 8, and conclude our work and sketch our future application in Sect. 9.

2 Related Work

Heterogeneous Information Network mining has gained traction for many applications in linked data analytics such as ontology, social network and knowledge graph [2] . We leverage RDD structure of Apache Spark in our proposed S-MPS algorithm for efficiently collecting path-instances of meta-paths in large HINs in a

distributed and parallel manner. One of the most common tasks in these applications is to measure similarity between vertices in HIN. The seminal work at [1] proposes Path-Sim, an algorithm to measure similarity between two vertices from the number of path- instances of a specific meta-path between these objects. Rather than number of path- instances, there are other similarity measures utilizing more information in HIN as Path-Constrained Random Walks [20], HeteSim [21], W-PathSim [4], etc. For instance, W-PathSim utilizes the content information of vertices (such as paper abstract of (Paper)-typed vertices in DBLP) to calculate the similarity between vertices of the same type. In DBLP, W-PathSim is calculated based on the topic similarity of two (Paper) vertices (or paper for brevity) in path-instances of a given meta-path. Each paper is represented by a topic vector which is generated by using LDA topic modeling algorithm [22]. Besides the tradition approach of relying on domain experts to manually specify meta-paths, there have been several studies about automatic algorithms to discover meta-paths in HINs. In [11], the authors propose FSPG, an algorithm based on Breadth-First Search to find all meta-paths of a given HIN. In [23], the authors propose algorithms to compute the top-k shortest paths from one set of target vertices to another set of target vertices in a graph [1]. A common aspect of these studies is that the proposed solutions are able to be executed with an input HIN fit to the memory of one computer. These solutions can be restricted in contexts of large HINs with millions of vertices and edges such as DBLP, YAGO, FreeBase. To mitigate this restriction, we propose distributed algorithms which can work with large HINs. Our proposed algorithms are developed on Apache Spark, a well-known distributed computing framework. Apache Spark was developed in 2009 in UC Berkeley's AMPLab by Matei Zaharia to process huge data. Apache Spark is based on Hadoop Map-Reduce Model. Apache Spark runs on Spark Cluster which contains one master and several workers. Apache Spark can divide the huge data into several parts. Each part is moved to the worker nodes. The Map-Reduce Model is used to process data parts in each worker. Apache Spark has Resilient Distributed Data Set (RDD) structure which can help to divide huge data and distribute data parts to worker nodes to execute in parallel. We leverage RDD structure in our proposed S-MPS algorithm for efficiently processing path-instances of meta-paths in large HINs in a distributed manner. Moreover, we use the existing meta-paths between two vertices to choose the adjacency vertices in the path instances of existing meta-paths instead of choosing all adjacency vertices as in D-BFS. With this filter operation, we can reduce the execution time for discovering the shortest meta-path between two vertices.

3 Preliminaries

Definition 1 An information network is defined as a directed graph G = (V, E) with a vertex type mapping function φ: V $\rightarrow$ A and an edge type mapping function ψ: E $\rightarrow$ R. Each object v $\in$ V belongs to one particular vertex type set A: φ (v) $\in$ A, and each link e $\in$ E belongs to a particular edge type in the edge type set R: ψ(e) $\in$ R.

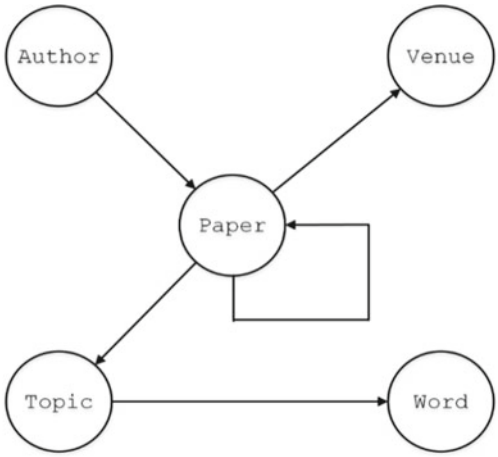


Fig. 1 Network schema of DBLP

The information network is called heterogeneous information network if the vertex types |A|>1 or the edge types |R|>1; otherwise, it is a homogeneous information network [2].

Example 1 DBLP is a typical HIN with the following vertex types: (Author), (Paper), (Venue), (Topic), (Word). (Word)-typed vertices are words extracted from the abstract section of a paper. From these (Word)-typed vertices, (Topic)-typed vertices are discovered using the LDA topic modeling algorithm [22]. The edge types of DBLP are [Write], [PublishedAt], [Mention], [Contain] and [Cite]. Figure 1 depicts a network schema of DBLP.

Definition 2 The network schema of a HIN, denoted as TG = (A, R), is a meta template for a HIN G = (V, E) with the vertex type mapping $\varphi : V \rightarrow A$ and the edge type mapping ψ: E→R, which is a directed graph defined over object types A, with edge as relations from R [2].

Definition 3 A meta-path P is a path defined on the network schema TG = (A,R) of a given HIN, and is denoted in the form of $A_1 \xrightarrow{R1} A_2 \xrightarrow{R2} \ldots \xrightarrow{Rl} A_{l+1}$, which defines a composite relation R = R1° R2° ... ° Rl between type A_1 and A_{l+1}, where ° denotes the composition operator on relations [1].

Example 2 Some meta-paths of DBLP are listed as follows:

Meta-path 1: (Author) $\xrightarrow{\text{Write}}$ (Paper). This meta-path expresses the author writing a paper.

Meta-path 2: (Author) $\xrightarrow{\text{Write}}$ (Paper) $\xrightarrow{\text{Mention}}$ (Topic). This meta-path expresses the author writing a paper and the paper mentions a topic.

Meta-path 3: (Author) $\xrightarrow{\text{Write}}$ (Paper) $\xrightarrow{\text{Mention}}$ (Topic) $\xrightarrow{\text{Contain}}$ (Word) . This meta-path expresses the author writing a paper, then the paper mentions a topic and the topic contains words.

Table 1 Some of the shortest meaningful meta-paths from source to target vertex of DBLP

	Source vertex	Destination vertex	Shortest meaningful meta-paths	Support
1	Paper	Word	Paper → Topic → Word	1,692,444
2	Author	Venue	Author → Paper → Venue	308
3	Author	Topic	Author → Paper → Topic	78,440
4	Paper	Word	Paper → Topic → Word	1,692,444

Two important concepts related to meta-path are meta-path length and path instance.

Definition 4 Given a meta-path $A_1 \xrightarrow{R1} A_2 \xrightarrow{R2} ... \xrightarrow{Rl} A_{l+1}$, we define length of a meta-path is the number of relations of this meta-path.

From Example 2, we can see that meta-path 1 has length 1, meta-path 2 has length 2 and meta-path 3 has length 3.

Definition 5 A path-instance of a specific meta-path $A_1 \xrightarrow{R1} A_2 \xrightarrow{R2} ... \xrightarrow{Rl} A_{l+1}$ is a path p of HIN where p: $a_1 \xrightarrow{R1} a_2 \xrightarrow{R2} ... \xrightarrow{Rl} a_{l+1}$ and a_i is a vertex of HIN, $a_j \xrightarrow{R_j} a_{j+1}$ is an edge with source vertex a_j and target vertex a_{j+1} of HIN.

In real HIN, some meta-paths do not have path instance. For example, in a subset of DBLP, we consider two meta-paths (Author)→(Paper) and (Paper)→(Venue). Suppose that, meta-path (Author)→(Paper) contains two path-instances such as (a_1)→(p_1) and (a_2)→(p_1). Similarly, meta-path (Paper)→(Venue) contains two path-instances such as (p_3)→(v_1) and (p_4)→(v_1). But we don't have any path-instance of meta-path (Author)→(Paper)→(Venue). This is the reason why we define the meaningful meta-path.

Definition 6 The support of a meta-path P, denoted by support(P), is the number of path instances of this meta-path.

Definition 7 A meta-path P is called a meaningful meta-path if support(P) > 0.

Definition 8 Given two vertices s and t of HIN, shortest meaningful meta-path p from s to t is a meaningful meta-path and the length of this meta-path is shortest in all meaningful meta-paths between vertex s and vertex t.

Example 3 Some shortest meaningful meta-paths of length 2 of DBLP are shown in Table 1.

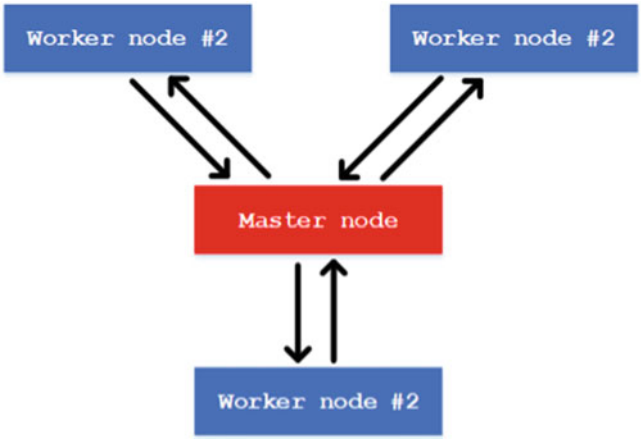


Fig. 2 The spark cluster with one master node and three worker nodes

4 Distributed Computing with Apache Spark

4.1 Apache Spark

Apache Spark is a software system to process big data on distributed environment with many computers [17]. Apache Spark was initially started by Matei Zaharia at University of Berkeley's AMPLab in 2009. Map-Reduce Model is a framework for distributed processing of big data.

Let D be a large data set, we divide D into many parts E_1, E_2, ..., E_n. Each part E_i is moved to worker node W_i. The Map-Reduce Model is used to process part E_i at worker node W_i and reduce the results of worker nodes at Master nodes as shown in Fig. 2.

4.2 The Map-Reduce Model

In this section, we present the Map-Reduce Model of Distributed Computing. The Map-Reduce Model is a programming model for processing large data set based on the "divide-and-conquer" strategy. First, a huge data is divided into many parts. These data parts are subsequently distributed to the worker nodes. A function f is provided to a worker node to process the data part at that node. The execution of f can be handled separately in each worker node. A typical Map-Reduce program works in two phases: map phase and reduce phase.

Suppose that, we would like to count the number of occurrences of words in a document that contains three lines as follows:

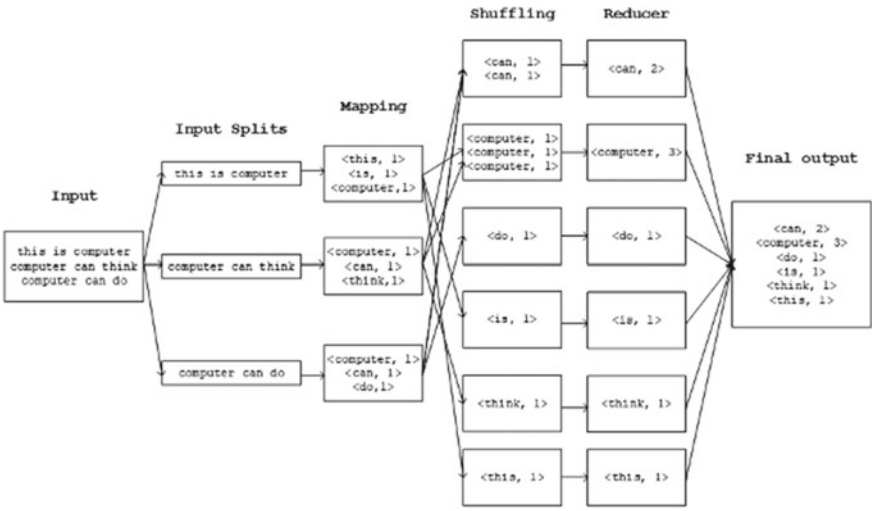


Fig. 3 Map Reduce model of the distributed computing

This is a computer.
Computer can think.
Computer can do.

In Fig. 3, these lines are divided into 3 parts. Each part contains one line and is moved to the worker nodes as local data. In the map phase, data in each worker node is counted. In the first line of our example, the function counts the number of occurrences of each word in data part of worker node and generates a tuple (word, number of occurrences) such as (this,1), (is,1) and (computer,1). In the shuffling phase, the tuples with form (word, 1) are sorted, we have (can, 1), (can, 1), etc. In the reduce phase, tuples of the same word are summarized. Finally, we get these following tuples indicating word counts in the given document: (can, 2), (computer, 3), (do, 1), (is, 1), (think, 1) and (this, 1). Because the huge data is processed in parallel the execution time is reduced significantly.

5 The D-BFS Algorithm for Shortest Meaningful Meta-Path Discovery

Problem Definition: Let HIN be a large graph G=(V, E). Each vertex $v \in V$ has vertex type in L(V) and each edge $e \in E$ has a label name in L(E). Let s, t be two vertices of G, find all the shortest meaningful meta-path connecting vertex s to vertex t.

D-BFS algorithm on Apache Spark is shown in Algorithm 1. In Apache Spark, the graph is represented by two tables. The Vertex table contains all vertices and

Table 2 Edge table is divided into many small parts then moved to the worker nodes

Worker node 1	Worker node 2	Worker node 3
(v3,v4)	(v3,v10)	(v3,v18)
(v3,v6)	(v3,v12)	(v3,v20)
(v3,v7)	(v4,v10)	(v5,v10)
(v4,v7)	(v4,v12)	(v5,v12)
(v4,v8)	(v4,v12)	(v5,v12)
(v4,v9)	(v4,v25)	(v5,v25)

Table 3 The final result at master node

Extended Path	From Worker
(v2,v3,v4)	Worker 1
(v2,v3,v6)	Worker 1
(v2,v3,v7)	Worker 1
(v2,v3,v10)	Worker 2
(v2,v3,v12)	Worker 2
(v2,v3,v18)	Worker 3
(v2,v3,v20)	Worker 3

Edge table contains all edges of HIN. In Apache Spark, Edge Table is divided into small parts and moved to worker nodes as shown in Table 2.

We start with path that contains edge (v2,v3). This path is extended by concatenating edge (v2,v3) with the edges in local data of each worker node. This concatenation can be processed in parallel at worker nodes. The edge (v2,v3) is joined in parallel with edges in the local data of each worker node. The master node will summarize the result paths from worker nodes. Table 3 contains the extended paths of each worker node.

The D-BFS algorithm is shown in Algorithm 1.

The function **EdgeFilter(Ej,ET)** in line 2 of the Algorithm 1 will expand the path of ET (result path) by joining with the edge in local part E_j of each worker node j. Function **GetTargetVertex(p)** in line 6 returns the target vertex of path p. The path is extended if the target vertex of path p equal to the source vertex of edge e in local part E_j. Function **BuildNewPath(p, e.src)** in line 9 builds a new path by concatenating path p with the new edge e at the end of path p.

After finding all the paths between two vertices, we choose the shortest paths of two given vertices of HIN, and then we build the shortest meaningful meta-paths from these paths by using the vertex types and links of these shortest paths.

Algorithm 1 D-BFS algorithm

Input HIN as Graph G=(V,E); SrcVert: source vertex; TgtVert: target vertex
Output List of all shortest paths between SrcVert and TgtVert

```
1: function D- BFS(G, SrcVert, TgtVert)
2:    function EDGEFILTER(E_j, ET)          ▷ This function will be mapped to worker nodes.
                                            E_j is the set of edges in worker node j and ET is the set of paths of Graph needed to expand
3:       NewPath ← []
4:       for each path p in ET do
5:          dp ← GetTargetVertex(p)          ▷ Get the target vertex of path p
6:          for each edge e in E_j do
7:             if e.src = dp then
8:                pnew ← BuildNewPath(p, e)
9:                NewPath.append(pnew)
10:            end if
11:         end for                          ▷ edge e
12:      end for                             ▷ path p
13:      Divide set of E into k parts E_1, E_2, ..., E_k and move to the worker nodes
14:      ResultPath ← Choose all edges of E where source vertex of edge equal SrcVert
15:      ET ← ResultPath
16:      while 1 do
17:         Activate EdgeFilter(E_j,ET) at each worker   ▷ Distributed Processing at worker nodes
18:         Reduce NewPath of each worker nodes and summarize to FinalET at master node.
19:         if FinalET = ET then             ▷ all paths of graph have been checked
20:            break
21:         end if
22:         ET ← FinalET
23:      end while
24:      FinalResultPath ← []
25:      for each p of FinalET do
26:         if GetTargetVertex(p) = TgtVert then
27:            FinalResultPath.append(p)
28:         end if
29:      end for
30:      return FinalResultPath
31:   end function
32: end function
```

The disadvantage of the D-BFS algorithm is to collect all the adjacency vertices of a current vertex. D-BFS algorithm does not care the edge label of edge that contains the current vertex to adjacency vertex. In our application, we consider only the adjacency vertices that is contained in the path instances of specified meta-paths. Because of this drawback, the execution time of D-BFS is time consuming. We will improve the efficiency by developing the novel algorithm named S-MPS algorithm.

6 The S-MPS Algorithm for Discovering the Shortest Meaningful Meta-Path Discovery

Our novel S-MPS algorithm contains three steps:

- **Step 1:** Generating the network schema of given HIN.
- **Step 2:** Collecting the shortest meta-paths between two vertices of network schema. These meta-paths are called possible shortest meta-paths.
- **Step 3:** Verify each meta-path of possible shortest meta-paths to confirm whether it is meaningful or not. The result of Step 2 is the set of shortest meaningful meta-paths.

We will explain each above step as follows:

Step 1: Generating the network schema.

The network schema of a HIN is easy to build. First, we scan the Vertex table to collect all the vertex types of the vertices of HIN. Each vertex type is a vertex of a network schema. Let **VertexTypeSet** be the set of all vertex types of a given HIN. We discover the network schema by using Algorithm 2 as follows:

Algorithm 2 Generating the network schema of a given HIN G=(V, E)

Input HIN G=(V, E) and **VetexTypeSet** is a set of all vertex types of HIN
Output List of all meta-paths of HIN G=(V, E). This list identifies the network schema of HIN and denoted by NS(HIN)

```
1: MetaPathList ← []
2: for s in VetexTypeSet do
3:     for t in VetexTypeSet do
4:         if s ≠ t then
5:             Generate meta-path p: (s) — [e]→(t)
6:             NumPathInstance ← CalculateThePathInstance(p)
7:             if NumPathInstance > 0 then          ▷ p is a meaningful meta-path
8:                 MetaPathList.append(p)           ▷ save p
9:             end if
10:        end if
11:    end for
12: end for
```

We use Algorithm 2 to generate all the path instances of meta-path of length 1 such as the meta-path (s) — [e]→(t). The condition **NumPathInstance** $>$ **0** in line 7 will check meta-path p is a meaningful meta-path.

Step 2: Finding the shortest meta-paths between two vertices of network schema.

We scan the network schema to find all the shortest meta-path between vertex s and vertex t. These two vertices represent two vertex types of HIN.

Normally, the size of network schema is not big. We use traditional algorithms such as Dijkstra's algorithm or Bellman Ford's algorithm to find all the shortest meta-path between two vertices of network schema. Table 4 contains some shortest meta-paths between two vertices of DBLP network schema.

Table 4 Some shortest meta-paths between 2 vertices of DBLP's network schema

	Source Type	Target Type	Meta-paths	Length
1	Topic	Word	Topic → Word	1
2	Paper	Topic	Paper → Topic	1
3	Paper	Paper	Paper → Paper	1
4	Paper	Venue	Paper → Venue	1
5	Author	Paper	Author → Paper	1

Step 3: Verify each meta-path of possible meta-paths to confirm whether it is meaningful or not. The meaningful meta-paths are the result of shortest meaningful meta-paths.

From step 2, we know all possible shortest meta-paths between two vertices of network schema or two vertex type of HIN. We need to verify whether this shortest meta-path is meaningful or not. To solve this problem, we use the following principle.

(i) We generate all the meta-path of length 1 from possible shortest meta-paths of step 2. For example, we have Author → Paper → Topic → Word, the meta-paths of length 1 are Author → Paper, Paper → Topic and Topic → Word.
(ii) We verify whether each meta-path of length 1 is meaningful or not. If all meta-paths of length 1 are meaningful, then this meta-path is the potential meaningful meta-path and we continue to will verify the shortest meta-paths of length 2. For example: because Author → Paper, Paper → Topic and Topic → Word are meaningful, we continue to verify meta-path potential meaningful meta path of length 2 such as Author → Paper → Topic, Paper → Topic → Word. We continue this task until we reach a non-meaningful meta-path (meta-path without any path-instance).

The details of these steps are as follows:

6.1 Verify the Meaningful Meta-Path of Length 1 by Using the Map Reduce Model

Figure 4 illustrates a HIN with the meaningful meta-paths such as meta-path P1: (Author) → (Paper) with support(P1) = 5 and meta-path P2: (Paper) → (Venue) with support(P2) = 3.

Set E of HIN edges is divided into n small parts $E_1, E_2, ..., E_{k4}$. Each part is moved to the worker node. At worker node j, the edge of part E_j will be scanned and select the edge satisfying the given meta-path and generate the result of worker node j denoted by WR_j. This operation is executed in parallel at each worker node. When this operation is finished, the result WR_j of worker node j where j = 1 ... k is moved to the master node for generating the final result.

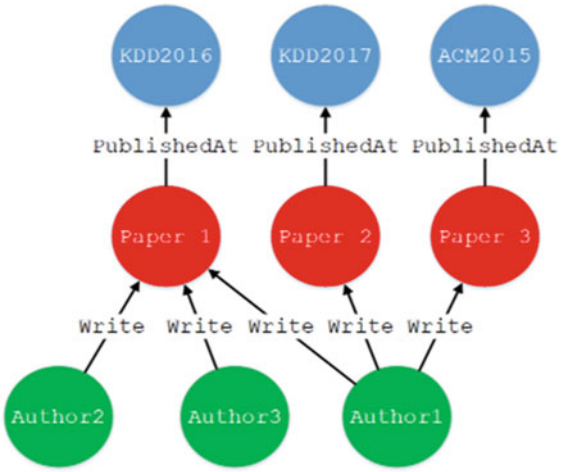


Fig. 4 An example of meta-paths of DBLP

We design Algorithm 3 as function named **DistributedGenPI_1(G,p)**.

Algorithm 3 Generating path instances of a given 1-length meta-path on Spark

Input The HIN denoted as G=(V,E) with a given 1 – length meta-path p where p: (a)—[rel]→(b)
Output A list of all path instances of a given meta-path

```
1: function DistributedGenPI_1(G, p)
2:     function WorkerGeneratePI_1(E_j, p)
3:         AnswerList ← []
4:         for each edge e in E_k do
5:             if e.label = "rel" then AnswerList.append((a,rel,b))
6:             end if
7:         end for
8:         return AnswerList
9:     end function
10:    Divide set of E into k parts E_1, E_2, ..., E_k and move to the worker nodes
11:    Triger tasks of workers to execute WorkerGeneratePI_1(Ej,p) where j=1...k in parallel and
       to receive WR_j                                          ▷ Parallel Processing
12:    Reducing the WR_j where j=1...k to FinalResult (a, rel, b)
13:    return FinalResult
14: end function
```

Function **WorkerGeneratePI_1()** in line 2 of Algorithm 3 will count the path-instances of a given meta-path of length 1. Since many worker nodes can process **WorkerGeneratePI_1(E_j,p)** in parallel, the execution time will be reduced significantly compared with the single machine model. The collaboration of master node and 3 worker nodes to generate path instances of meta-paths of length 2 (line 12) is shown in Fig. 5.

Example 4: This example shows the operation of Algorithm 1 to generate the path instance of meta-path as (Author) → (Paper). This is a meta-path of length 1.

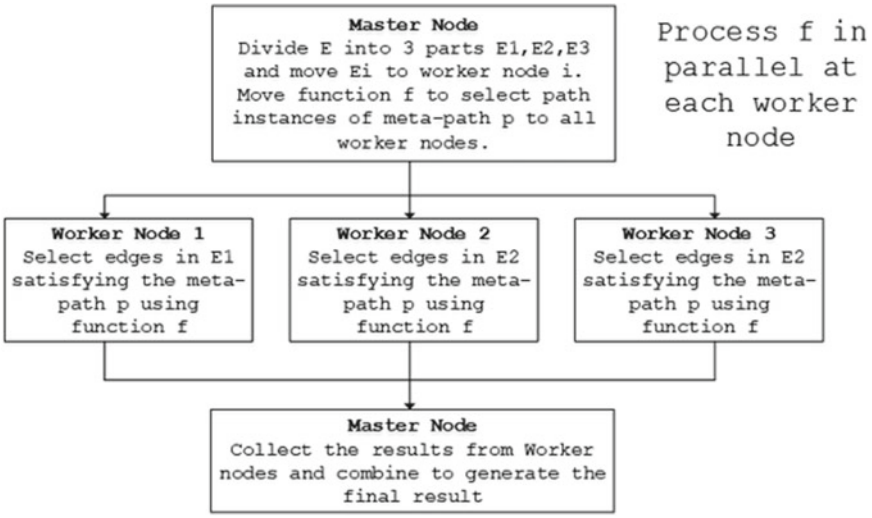

Master Node
Divide E into 3 parts E1,E2,E3 and move Ei to worker node i. Move function f to select path instances of meta-path p to all worker nodes.
Process f in parallel at each worker node
Worker Node 1
Select edges in E1 satisfying the meta-path p using function f
Worker Node 2
Select edges in E2 satisfying the meta-path p using function f
Worker Node 3
Select edges in E2 satisfying the meta-path p using function f
Master Node
Collect the results from Worker nodes and combine to generate the final result

Fig. 5 Collaboration of master node and worker nodes to generate path instances of meta-paths of length 1

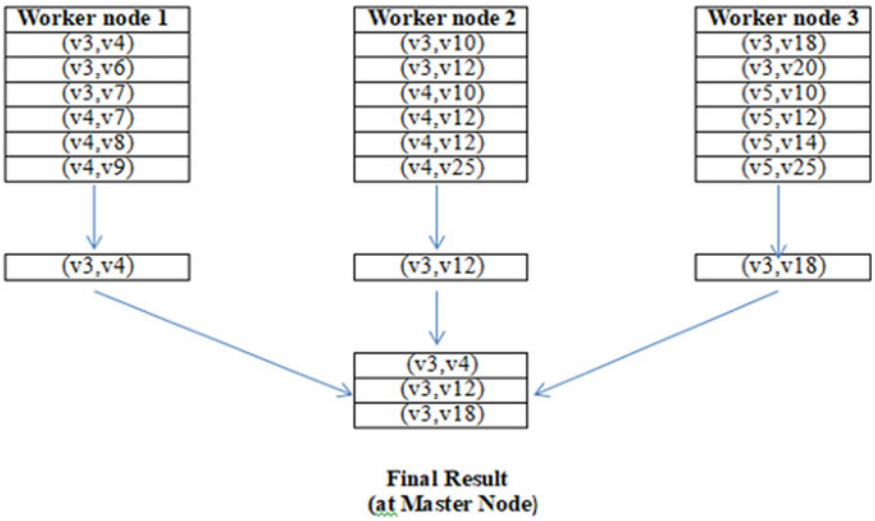


Fig. 6 Local processing with filter at each worker node and summarizing at master node

Suppose that the type of vertex v3 is "author" and types of vertices v4, v12, v18 are "papers". The remaining vertices have vertex types different from type "author" and type "paper". Then the filtered result at each worker node is shown in Fig. 6.

6.2 *Check the Meaningful Meta-Path of Length K by Using the Map Reduce Model*

The path instances of meta-path of length k are generated by using the Map-Reduce Model. Typical meta-paths of length 2 are as follows: p: (a) → [e1] → (b) → [e2] → (c).

We separate this meta-path into two meta-paths.

We have two meta-paths of length 1 as (a) → [e1] → (b) and (b) → [e2] → (c).

There are 2 steps for this operation. In the first step, we use the Distributed-GenPI_1(G,p) where p: (a) → [e1] → (b) to generate the relation R(a,e1,b).

For the second step, we scan R(a,e1,b). For each row r of R, we collect the edge e of HIN with e.label = r.e1 and e.tgt (target vertex) = r.b.

We divide the set E of HIN into k small parts. For each part, we choose edge e in this part where e.label = r.e1 and e.tgt = r.b.

Similarly, we generate the path instances of meta-path of length k by using the distributed algorithm.

We list a typical meta-path of length k where k=3 as follows:

$$(a) \rightarrow [e1] \rightarrow (b) \rightarrow [e2] \rightarrow (c) \rightarrow [e3] \rightarrow (d)$$

We use the above method to get R(a,e1,b,e2,c), then we divide the set E of HIN into k parts. For each part, we choose edge e in R where e.label=r.e3 and e.tgt = r.c.

This operation runs in parallel at each worker node, so the execution time will be reduced significantly.

Some shortest meaningful meta-path between 2 different-type vertices of DBLP are shown in Table 5.

7 Comparing the Performance of the D-BFS and the S-MPS Algorithms

In this section, we analyze the performance of the D-BFS algorithm and the S-MPS algorithm.

Example 4 In Apache Spark environment, edges of graph are stored in a table. They are shown in Table 6.

With the D-BFS algorithm, we join two Edge Tables to build edges of Breadth-First Tree as shown in Table 7 and Fig. 7.

One tuple of Edge Table 1 is joined with each tuple of Edge Table 2. The execution time is needed for joining 6 tuples. Edge Table 3 is the joining result and contains six tuples.

In S-MPS algorithm, we knew all meta-paths between two vertices. We checked meta-path (Author) → (Paper) and we have path instance (v1) → (v2). Now we

Table 5 Some shortest meaningful meta-paths between 2 vertices with different types of DBLP

	Source type	Target type	Meta-paths	Length	Support
1	Topic	Word	Topic → Word	1	494
2	Paper	Topic	Paper → Topic	1	34,260
3	Paper	Paper	Paper → Paper	1	10,234
4	Paper	Venue	Paper → Venue	1	126
5	Author	Paper	Author → Paper	1	7,844
6	Paper	Word	Paper → Topic → Word	2	1,692,444
7	Author	Paper	Author → Paper → Paper	2	24,224
8	Author	Venue	Author → Paper → Venue	2	308
9	Author	Topic	Author → Paper → Topic	2	78,440
10	Author	Word	Author → Paper → Topic → Word	3	3,874,936

Table 6 Some shortest meaningful meta-paths between 2 vertices with different types of DBLP

Source vertex	Target vertex
v1	v2
v2	v3
v2	v4
v2	v5
v2	v6
v2	v7
v2	v8

Fig. 7 Breadth-first tree with different types of vertices

Table 7 Join operation of D-BFS algorithm

(a) Edge Table 1

Source	Target
v1	v2

(b) Edge Table 2

Source	Target
v2	v3
v2	v4
v2	v5
v2	v6
v2	v7
v2	v8

(c) Edge Table 3

Vertex1	Vertex2	Vertex3
v1	v2	v3
v1	v2	v4
v1	v2	v5
v1	v2	v6
v1	v2	v7
v1	v2	v8

Table 8 Joining operation of SMPS algorithm

(a) Edge Table 1

Source	Target
v1	v2

(b) Edge Table 2

Source	Target
v2	v8

(c) Edge Table 3

Vertex1	Vertex2	Vertex3
v1	v2	v8

check meta-path (Paper) → (Venue). Therefore, we focus on vertex v8, because vertex v8 is a Venue. We keep only the row of table satisfying the condition of the path instance of the specified meta-path. In this case, we use meta-path (Author) → (Paper) → (Venue), thus we have only one tuple as shown in Table 8.

Edge Table 1 will be joined with Edge Table 2. In this case, only one tuple of Edge Table 1 joins with one tuple of Edge Table 2. Edge Table 3 is the joining result and contains one tuple. Obviously, the execution time for joining operation of S-MPS algorithm is faster than the execution time for join operation of D-BFS algorithm.

8 Experiment and Discussion

8.1 Spark Stand-Alone Cluster

We run our system on a Spark standalone cluster has 1 master node that also doubles up as a master and worker node (32 GB RAM, 4 cores) and 3 worker nodes (each with 16 GB RAM 4 cores). We set up a Spark Cluster with HDFS of Hadoop Cluster as shown in Table 9.

Table 9 Configuration of spark cluster

Machine Name	IP Address	Role in Hadoop Cluster	Role in Spark Cluster
hadoop1	192.168.1.101	NameNode and DataNode	Master and Worker node
hadoop2	192.168.1.102	DataNode	Worker node
hadoop3	192.168.1.103	DataNode	Worker node
hadoop4	192.168.1.104	DataNode	Worker node

Table 10 Different length meta-paths

	Meta-path	Meta-path length
1	(Author) → (Paper)	1
1	(Author) → (Paper) → (Venue)	2
1	(Author) → (Paper) → (Topic) → (Word)	3
1	(Author) → (Paper) → (Paper) → (Paper) → (Paper)	4

8.2 Data Set

A subset of the DBLP citation network contains 100,000 vertices and 700,000 edges is used for testing our proposed system.

8.3 Comparison of the Execution Time Between S-MPS Algorithm and D-BFS Algorithm with Different Length Meta-Paths

We compare S-MPS algorithm and D-BFS algorithm with different meta-path lengths. We use the following meta-paths with different length meta-paths as shown in Table 10. We use Spark cluster system as mentioned above. The execution time is shown in Fig. 8.

Figure 8 illustrates that the execution time of S-MPS algorithm is faster than D-BFS algorithm with different length meta-paths.

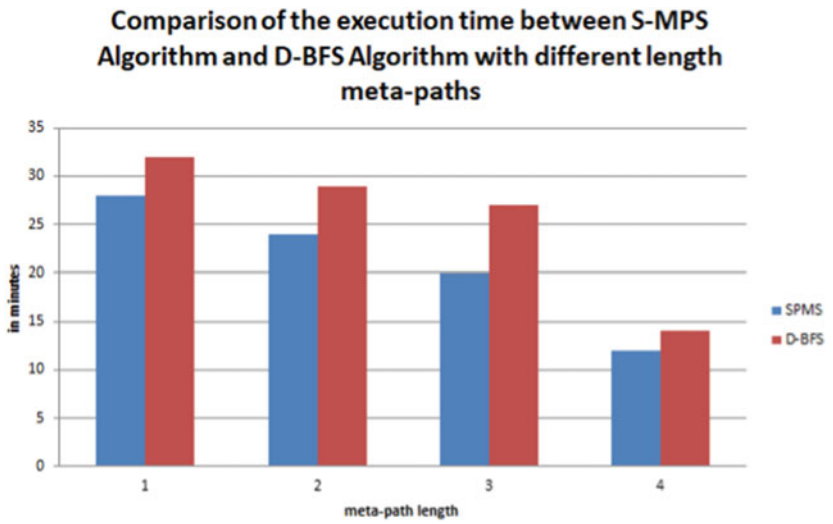


Fig. 8 Collaboration of master node and worker nodes to generate path instances of meta-paths of length 1

8.4 Comparison of the Execution Time Between S-MPS Algorithm and D-BFS Algorithm with Different Numbers of Worker Nodes

We conduct the experiment to measure the execution time of finding the shortest meaningful meta-paths between 2 vertices of large HIN on local environment and Spark Cluster by using S-MPS algorithm and D-BFS algorithm with meta-path (Author) → (Paper) → (Venue). The number of workers is changed. We hold that the execution time of S-MPS is reduced compared with D-BFS algorithm because with S-MPS, only the branches from one particular vertex satisfying the path instance of a specified meta-path are considered while in D-BFS all the possible branches from one particular vertex are considered and the time for considering all possible branches are time consuming. Figure 9 illustrates the comparison of execution time of S-MPS algorithm and D-BFS algorithm on local environment and Spark Cluster with different numbers of worker nodes.

This experiment proves the performance of S-MPS compared with D-BFS and the Spark Cluster compared with Local environment and the changes of number of worker nodes in Spark cluster. This experiment proves the performance of distributed computing on Spark Cluster for processing big data with 2 worker nodes to 4 worker nodes.

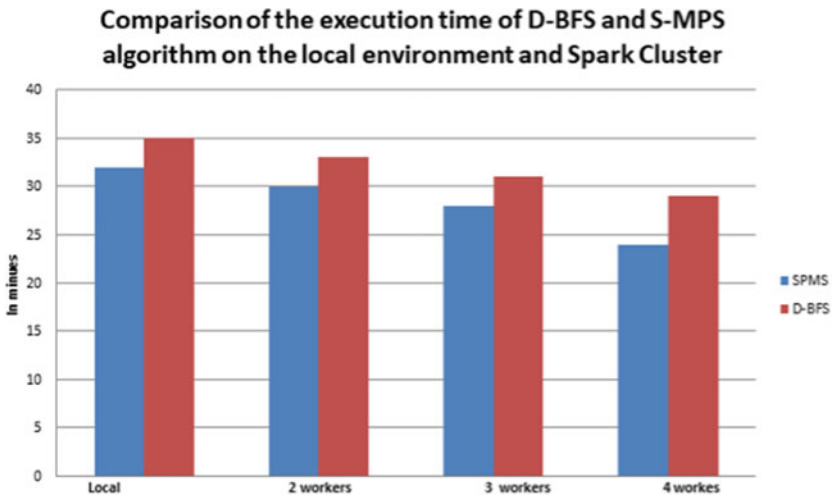


Fig. 9 Comparison of the execution time of D-BFS and S-MPS algorithm on the local environment and spark cluster

9 Conclusions

In this chapter, two distributed algorithms D-BFS and S-MPS algorithm are proposed to discover all the shortest significant meta-paths between two vertices of secured large HIN. We use Spark distributed environment with Map-Reduce Model to develop these algorithms for processing large HIN. In Spark distributed environment, HIN graph is represented by two tables, one table for vertices and another table for edges. The edges of graph are processed in parallel at each worker node of Spark Cluster. We implemented D-BFS algorithm in Apache Spark by joining and extending the meta-path in Breadth-First Tree. We implemented S-MPS algorithm bases on the network schema and path-instance to generate shortest meaningful meta-paths with the filter of the path instances of the specified meta-path. We conducted experiments for comparing the execution time of D-BFS algorithm and S-MPS algorithm with DBLP. The experiment result proves the performance of S-MPS algorithm compared with D-BFS algorithm. We plan to use shortest meaningful meta-paths in question answering system. Each meta-path will be as a template for generating multi-hop questions. The path instances of this shortest meaningful meta-path will be the answer of this question.

Acknowledgements This research is funded by Vietnam National University Ho Chi Minh City (VNU-HCMC) under the grant number DS2020-26-01.

References

1. Sun, Y., Han, J., Yan, X., Yu, P., Wu, T.: Path-Sim: meta path-based top-k similarity search in heterogeneous information networks. In: VLDB, pp. 992–1003 (2011). https://doi.org/10.14778/3402707.3402736
2. Shi, C., Li, Y., Zhang, J., Sun, Y., Yu, P.S.: A survey of heterogeneous information network analysis. IEEE Trans. Knowl. Data Eng. (2017). https://doi.org/10.1109/TKDE.2016.2598561
3. Phan, T., Do, P.: Building a Vietnamese question answering system based on knowledge graph and distributed CNN. Neural Comput. Appl. **33**, 14887–14907 (2021). https://doi.org/10.1007/s00521-021-06126-z
4. Do, P., Pham, P.: DW-PathSim: a distributed computing model for topic-driven weighted meta-path-based similarity measure in a large-scale content-based heterogeneous information network. J. Inf. Telecommun. **3**(1), 19–38 (2019). https://doi.org/10.1080/24751839.2018.1516714
5. Salhi, D., Tari, A., Kechadi, T.: Using clustering for forensics analysis on internet of things. Int. J. Softw. Sci. Comput. Intell. (2021)
6. Kong, X., Cao, B., Yu, P., Ding, Y., Wild, D.: Meta path-based collective classification in heterogeneous information. Networks (2013). https://doi.org/10.1145/2396761.2398474
7. Trappey, A.J., Trappey, C.V., Chang, A., Li, J.X.: Deriving competitive foresight using an ontology-based patent roadmap and valuation analysis. In: International Journal on Semantic Web and Information Systems, pp. 68–91 (2019). https://doi.org/10.4018/IJSWIS.2019040104
8. Ho, T., Do, P.: Discovering communities of users on social networks based on topic model combined with Kohonen network. In: Seventh International Conference on Knowledge and Systems Engineering, pp. 268–273 (2015). https://doi.org/10.1109/KSE.2015.54.
9. Do, P.: A system for natural language interaction with the heterogeneous information network. In: Handbook of Research on Cloud Computing and Big Data Applications in IoT (2019)
10. Besmir, S., Florie, I., Lule, A.: Integration of semantics into sensor data for the IoT: a systematic literature review. In: International Journal on Semantic Web and Information Systems (2020). https://doi.org/10.4018/IJSWIS.2020100101
11. Meng, C., Cheng, R., Maniu, S., Senellart, P., Zhang, W.: Discovering meta-paths in large heterogeneous information networks. In: Proceedings of the 24th International Conference on World Wide Web (2015). https://doi.org/10.1145/2736277.2741123
12. Liu, H., Jin, C., Yang, B., Zhou, A.: Finding Top-k shortest paths with diversity. In: IEEE Transactions on Knowledge and Data Engineering, pp. 488–502 (2018). https://doi.org/10.1109/TKDE.2017.2773492.
13. Khekare, G., Verma, P., Dhanre, U., Raut, S., Sheikh, S: The optimal path finding algorithm based on reinforcement learning. In: International Journal of Software Science and Computational Intelligence (2020). https://doi.org/10.4018/IJSSCI.2020100101
14. Iqbal, S., Hussain, I., Sharif, Z., Qureshi, K.H., Jabeen, J.: Reliable and energy-efficient routing scheme for underwater wireless sensor networks (UWSNs). In: International Journal of Cloud Applications and Computing (IJCAC) (2021). https://doi.org/10.4018/IJCAC.2021100103
15. Zhu, Z., Cheng, R., Do, L., Huang, Z., Zhang, H.: Evaluating Top-k meta path queries on large heterogeneous information networks. In: IEEE International Conference on Data Mining, pp. 1470–1475 (2018). https://doi.org/10.1109/ICDM.2018.00204
16. Drabas, T., Lee D.: Learning PySpark. Packt (2017)
17. Al-Nawasrah, A., Almomani, A.A., Atawneh, S., Alauthman, M.: A survey of fast flux botnet detection with fast flux cloud computing. In: International Journal of Cloud Applications and Computing (2021). https://doi.org/10.4018/IJCAC.2020070102
18. Dave, A., Jindal, A., Liy, L.E., Xin, R., Gonzalez, J., Zaharia, M.: GraphFrames: an integrated API for mixing graph and relational queries. In: Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems (2016). https://doi.org/10.1145/2960414.2960416

19. Koji, U., Toyotaro, S., Naoya, M., Katsuki, F., Satoshi, M.: Efficient breadth-first search on massively parallel and distributed-memory machines. In: Data Science and Engineering (2017). https://doi.org/10.1007/s41019-016-0024-y
20. Shi, C., Li, Y., Zhang, J., Sun, Y.: A survey of heterogeneous information network analysis. IEEE Trans. Knowl. Data Eng. (2017)
21. Ni, L., William, C.: Fast query execution for retrieval models based on path-constrained random walks. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2010). https://doi.org/10.1145/1835804.1835916
22. Chuan, S., Xiangnan, K., Yue, H., Philip, Y., Bin, W.: HeteSim: a general framework for relevance measure in heterogeneous networks. In: IEEE Transactions on Knowledge and Data Engineering, vol. 26 (2013). https://doi.org/10.1109/TKDE.2013.2297920
23. Blei, D.M., Ng, A.Y., Michael, I.J.: Latent Dirichlet allocation. J. Mach. Learn. Res. (2003)
24. Lijun, C., Xuemin, L., Lu, Q., Jeffrey, X., Jian, P.: Efficiently computing Top-K shortest path join (2015)

Dr. Phuc Do is currently an Associate Professor of the University of Information Technology (UIT), VNU-HCM, Vietnam. His research interests include data mining, text mining, social network analysis, information network analysis, big data analysis and applications.

Creation of a Dataset Modeling the Behavior of Malware Affecting the Confidentiality of Data Managed by IoT Devices

Alberto Huertas Celdrán, Pedro Miguel Sánchez Sánchez, Fabio Sisi, Gérôme Bovet, Gregorio Martínez Pérez, and Burkhard Stiller

Abstract Internet-of-Things (IoT) platforms are vulnerable to cyberattacks due to their exposition to the internet and the resource-constrained capabilities of their devices. Cyberattacks can be conducted by different malware families, being spyware and backdoor, two of the most relevant due to their impact on data handled by IoT devices. Traditional detection systems based on signatures are unsuitable for zero-day attacks. Therefore, the current trend is to apply machine and deep learning to detect anomalies in the device behavior. However, there is a lack of datasets containing clean and infected behavioral data to train models. Most of the existing datasets do not consider resource-constrained devices, the malware samples are obsolete, and they do not monitor device behaviors. Thus, this work presents FabIoT, a dataset modeling the behavior of a resource-constrained device while being infected by three real backdoors exhibiting heterogeneous attack behaviors. The monitored device is a Raspberry Pi 3, acting as a spectrum sensor of a crowdsensing platform called ElectroSense. The dataset contains several hours of behavioral data sources such as CPU, I/O, Network, Memory, or Scheduler. Statistical analysis of the collected

A. Huertas Celdrán (✉) · F. Sisi · B. Stiller
Communication Systems Group CSG, Department of Informatics, University of Zurich UZH, CH-8050 Zürich, Switzerland
e-mail: huertas@ifi.uzh.ch

F. Sisi
e-mail: fabio.sisi@ifi.uzh.ch

B. Stiller
e-mail: stiller@ifi.uzh.ch

P. M. Sánchez Sánchez · G. Martínez Pérez
Department of Information and Communications Engineering, University of Murcia, 30100 Murcia, Spain
e-mail: pedromiguel.sanchez@um.es

G. Martínez Pérez
e-mail: gregorio@um.es

G. Bovet
Cyber-Defence Campus, armasuisse Science & Technology, CH-3602 Thun, Switzerland
e-mail: gerome.bovet@armasuisse.ch

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_11

data demonstrated the suitability of the dataset to detect normal and under-attack behaviors.

Keywords Malware · Dataset modeling · Behavior fingerprinting

1 Introduction

Over the last decade, there has been a steadily increasing demand for Internet of Things (IoT) devices and services. The combination of a large number of resource-constrained devices enables many scenarios such as smart cars and homes. Other possible large-scale applications include smart cities or fully automated factories. One promising IoT concept that has gained popularity in the last years is crowdsensing. Crowdsensing is a concept where a large number of individuals contribute small amounts of data to a more extensive collection. The goal of such a platform could, for example, could be monitoring the radio frequency spectrum. ElectroSense, a real IoT platform, does precisely that and senses the radio frequency spectrum using cheap and affordable hardware on a global scale.

As the application of IoT devices gains in popularity, there are also rising cybersecurity concerns [27]. Cyberattacks against IoT networks or platforms have been becoming increasingly common. Among these attacks, spyware and backdoor are two of the most impactful and recent ones [3]. Data is, in many cases, the goal of cyberattacks against crowdsensing or IoT platforms in general. While the data handled by IoT devices should be protected and confidential, with every passing year, attackers launch more and more spyware, and backdoor attacks. One possible reason for the increasing number of attacks is that IoT devices can be used as a gateway to attack other possibly more critical devices in the same network [17].

Traditional malware detection systems often apply static analysis. It means that these systems rely on a previously known trace or signature to detect malware [13]. Attackers can exploit this fact and use evasion techniques such as packing or obfuscation to change its signature and, therefore, make it harder to detect again [30]. Furthermore, that would imply that traditional systems are not capable of detecting new malware that has never been seen before, called zero-day attacks [13]. Opposed to static analysis, there is dynamic analysis, also called behavior fingerprinting [36]. The goal of dynamic analysis is to model the behavior of a device and use that model to detect anomalies produced by malware. The primary advantage of dynamic analysis is that while it is easy for attackers to change the malware signature, the malware behavior and causes on the device cannot be changed that easily. That also implies that it is possible to detect and mitigate zero-day attacks in certain circumstances [13].

Nowadays, solutions based on Machine and Deep Learning (ML/DL) are the most promising to apply dynamic analysis [36]. However, ML/DL models require datasets to be used successfully and efficiently. These datasets can model only the normal behavior of devices or the normal plus behavior when devices are affected

by cyberattacks. However, there are three main open challenges in terms of existing datasets. Firstly, most of the existing datasets do not concern themselves with resource-constrained devices but rather with other types of architectures such as cloud data centers or client-server structures. Another issue is that many datasets are too old and do not accurately model today's malware. In addition, recent approaches such as crowdsensing are often not considered in existing datasets. Lastly, most datasets fail to capture the devices internal behavior and instead focus on network-related activities.

To improve the previous challenges, the main contribution of this chapter is *FabIoT*, a dataset modeling the internal behavior of a resource-constrained device while being infected by different families of recent backdoor. The resource-constrained device is a Raspberry Pi acting as a radio frequency spectrum sensor of ElectroSense, a real-world IoT crowdsensing platform. Internal events belonging to heterogeneous categories such as CPU, memory, network, or scheduler were monitored when the Raspberry Pi was (i) attacked by three backdoor samples and (ii) no attack was happening. The backdoor samples have been selected after analyzing recent trends in malware development. To conclude, an statistical analysis has demonstrated the suitability of the created dataset to detect anomalies produced by the backdoor behaviors.

The remainder of this chapter is organized as follows. Section 2 analyzes existing work on backdoor, datasets, and detection solutions. While Sect. 3 introduces ElectroSense and motivates security concerns of spectrum sensors in general, Sect. 4 outlines the main steps to create the dataset. Section 5 presents the results demonstrating the suitability of the proposed dataset. Finally, Sect. 6 summarizes the work and outlines major findings.

2 Related Work

This section starts by introducing the main aspects of spyware and backdoor malware. Subsequently, it provides an analysis of existing datasets modeling malware behavior. Finally, the section finalizes with an examination of the current state of malware detection solutions using datasets.

2.1 Backdoor and Spyware Background

Backdoors are an access point to a system for third-party members without proper authentication. The general conception is that backdoors are primarily implemented maliciously, but that is not quite the truth. Many software companies include backdoors in their products to make troubleshooting and maintenance easier [20]. However, if the intent behind the implementation of a backdoor is malicious, there are certain traits that most backdoors share. One of those traits is stealth. Backdoors

should remain undetected by the user to be useful. The user should remain completely unaware that his system is compromised. Additionally, backdoors do not attack the system directly but act more like a gateway for other malicious activity. It is not rare for backdoors to be combined with different types of malware. A typical example is the combination of a trojan and a backdoor. As soon as the trojan is activated, e.g., executed by the victim, it installs a backdoor for the attacker to perhaps return at a later point in time. In general, a piece of malware can rarely be categorized in a single family of malware. Often, malware implementations exhibit and combine attributes from many different malware types.

Spyware malware samples, also called info stealers, steal sensitive information from the victim and send that data to the attacker. Among the most common stolen information, we can find credit card details and passwords. Spyware is often found on personal computers or mobile phones since its primary motivation is financial. Webhooks, screen recorders, and keyloggers are popular implementations of this type of malware [22]. Due to its stealthy nature, spyware often conceals itself as part of a regular download or installation, similar to adware. Smartphones or mobile devices often get infected with spyware if applications are downloaded from non-official app stores. Because of these facts, very few spyware implementations for IoT devices are freely available since most target personal computers.

2.2 Datasets Modeling Malware Behavior

This section analyzes datasets containing abnormal device behavior caused by some attack. Most datasets contain normal and abnormal behavior to identify and differentiate normal and abnormal behavior.

The literature on datasets modeling device behavior by collecting network-related activity is considerable. LITNET-2020 [12] is the first dataset in this review. It contains normal network flows and while under 12 different attacks. The attacks include DDoS, port scanning, and smurf attacks, among others. 85 different types of network flows were captured in an academic network setting on general computer systems and for ten months. In [17], the authors collected data on IoT devices while conducting similar attacks. The authors also included regular PC systems in the data collection setup and the IoT devices. That is because IoT datasets often ignore traffic between IoT devices and regular computer systems.

The authors of [2] created a dataset where they focused their efforts on creating an IoT dataset but also including Industrial Internet of Things (IIoT) devices. The dataset consists of telemetry data of IoT and IIoT devices, operating system logs, and network data of the IoT network itself. The eight attack scenarios included DDoS, ransomware, and backdoor attacks, among others. Another dataset containing similar malware attacks is the IoT-23 dataset [14]. The authors captured 23 scenarios, 20 of which are from infected Raspberry Pis. In each scenario, a different type of malware attack was executed using different botnets. During the attack, the authors captured network data as raw network packet (.pcap) files. For the additional three uninfected

scenarios, real-world IoT devices were used. An essential aspect of this dataset is the contrast between malicious and benign network traffic. Also, dealing with botnets, [23] created a dataset where a typical home IoT environment was monitored while being under simulated and real attacks. The simulated attacks consisted of port scanning, spoofing, and brute force attacks conducted by the Mirai botnet. While the attacks took place, the authors captured .pcap files. In total, 42 files were collected.

Hamza et al. [19] created the ACM SOSR 2019 dataset using a relatively novel approach. The detailed procedure is explained in Sect. 2.3. The dataset contains network flow counters while the devices were functioning normally and while under attack. The attacks consisted of network attacks such as DoS, flooding, and ARP spoofing. The authors of [34] collected traces from simulated IoT devices and created the DS2OS dataset. Among the simulated IoT devices, there were movement sensors, thermostats, or smartphones. The goal of this work and dataset lies in the microservices that IoT devices offer. N-BaIoT was created in a very similar manner [29]. The network traffic of nine commercial IoT devices was collected while being under attack from two different botnets, Mirai and BASHLITE, similar to [14, 23]. Each measurement contains 11 labels, ten of which were collected while being under attack and one modeling benign behavior.

In 2018, the CIC (Canadian Institute for Cyber Security) published [39] and with it two datasets, the CIC-IDS 2017 and the CIC-IDS 2018. Both datasets contain raw network traffic captures collected during cyberattacks and benign samples. The seven attack scenarios include Brute Force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and Infiltration attacks. The difference between the two datasets is that in the 2017 version, the network traffic was collected from 25 users, whereas in the 2018 version, the data was collected from 500 devices in total.

As more and more people use the Android operating system, more and more devices are infected by malware. To combat this growing threat, there are also Android-based datasets to detect malware, such as the Android Malware Dataset (CIC-AndMal2017) [26], where the authors monitored 80 different network traffic features of Android devices. The data was collected on real smartphones with 426 malicious and 5065 benign applications being installed. In 2019, the second part of this dataset, CIC-InvesAndMal2019, was published in [44] and contains additional data such as API calls and system log files. The malware samples were categorized in either adware, ransomware, scareware, or SMS malware. Also, in the context of Android malware, there is the Android Adware, and General Malware Dataset (CIC-AAGM 2017) [25]. Many of the authors were also involved with [44] since the CIC published both datasets. The difference between the two datasets is that CIC-AAGM 2017 focuses on adware and general malware and only contains network traffic captures. It contains raw and processed network captures of 1900 applications, 400 of which are considered malware. The malicious applications were installed on real smartphones like in [44].

NGIDS-DS [18] is another relevant dataset that contains network traffic captures and system logs of enterprise-critical infrastructures, such as storage, web, or email servers. The authors collected data in a standard scenario under the subsequent seven network attacks: Exploits, DoS, Worms, Generic, Reconnaissance, Shellcode, and

backdoors. The IXIA Perfect Storm tool, a commercial-scale security test hardware platform, was used as a test platform.

Finally, the remaining three datasets contain system calls data. The two ADFA (Australia Defense Force Academy) datasets are similar in many regards. They both contain system calls when the system runs in normal and under attack modes. The attacks include brute force attacks, web shells, poisoned executables, and exploits. The main difference is that ADFA-LD [11] is based on Linux systems, and ADFA-WD [10] on Windows systems. The University of New Mexico (UNM) dataset [9] was created in 1999 and contains roughly 500 kB of system call data. ADFA-WD contains 13.6 GB of system call data. The UNM dataset contains normal and under-attack attack behavior similar to the ADFA datasets. Among the considered attacks, trojans and buffer overflows are the most representative. Due to the technological advancements made during the last 22 years and its substantial age, the UNM dataset is now considered outdated.

Table 1 gives an overview of public datasets that try to model device behavior while being under attack. As can be observed, most datasets are focused on network-related metrics. It is common for datasets to model botnets such as Mirai or BASHLITE since they are among the most prominent and dangerous IoT malware. Also, as stated in [36], other metrics such as system calls or resource usage are under-used. The network-based datasets fail to capture the internal behavior of a device and only focus on communication. However, information about the internal behavior would be immensely useful to understand how malware works.

2.3 Malware Detection Solutions

The internal behavior of devices is a topic that has attracted more and more interest from the research community in recent years. Device fingerprinting aims to describe and model the internal behavior of a device. Sánchez et al. [36] summarizes the recent research in device fingerprinting. The authors concluded that most approaches use network-related activity to model the behavior of a device in terms of anomaly/attack detection. However, other promising dimensions such as system calls, the usage of resources, and HPCs were also researched and showed potential. Additionally, most solutions only consider basic attacks, missing sophisticated ones since they are rare and infrequent [10].

The authors of [41] proposed an approach where a combination of network activities and system calls was used to detect malware. Using this combination of behavior dimensions, the authors achieved a detection rate of 99.54% utilizing ML algorithms. In [1], a solution combining blockchain and ML in the context of IoT devices was developed. An ML algorithm is used to monitor and analyze the behavior of all devices in the network. The N-BaIoT dataset [29] was used in this work. Another solution considering botnets was presented in [16]. The idea was to detect an attack based on a trust score calculated for all traffic flows in an IoT network. The experiment was conducted in a software-defined networking (SDN) environment. Also,

Table 1 Datasets modeling malware behavior (grouped by behavior source, and sorted by year)

Name	Year	Device type	Behavior source	Behavior data	Tested malware
LITNET-2020 [12]	2020	General computers	Network	Processed network flows	Network attacks
IoT-Keeper [17]	2020	IoT	Network	Raw network flows	Network attacks
TON_IoT [2]	2020	IoT	Network/System Logs	Telemetry data, logs and network	9 different attack scenarios
IoT-23 [14]	2020	IoT	Network	Raw network capture	Botnets
IoT network intrusion dataset [23]	2019	IoT	Network	Raw network capture	Botnets
ACM SOSR 2019 [19]	2019	IoT	Network	Network flow counters	Network attacks
CIC-InvesAndMal [44]	2019	Android	Network/System Logs	Network features, API calls and logs	Mobile Malware
N-BaIoT [29]	2018	IoT	Network	Processed network flows	Botnets
DS2OS [34]	2018	IoT	Network	Application layer traces	Network attacks
CIC-IDS 2018 [39]	2018	General computers	Network	Raw and processed network capture	Network attacks
CIC-IDS 2017 [39]	2018	General computers	Network	Raw and processed network capture	Network attacks
CIC-AAGM [25]	2017	Android	Network	Raw and processed network capture	Adware and malware
NGIDS-DS [18]	2017	Critical Infrastructure	Network/System logs	Raw network capture/OS logs	Network attacks
ADFA-WD [10]	2014	General computers	System calls	DLL traces	60 different attack sets
ADFA-LD [11]	2013	General computers	System calls	System logs	60 different attack sets
UNM Dataset [9]	1999	General computers	System calls	System call identifiers	Trojans, buffer overflows and link attacks

in the context of SDN, the authors of [8] presented a system that monitors network traffic. Due to its nature, SDN is arguably more susceptible to DDoS or Man in the Middle attacks. The multi-stage system managed to detect and diminish DDoS and port scan attacks.

Hamza et al. proposed a system in [19] where the goal is to identify abnormal behavior in IoT devices using the manufacturer usage description (MUD). The MUD essentially defines the normal behavior of a device and is provided by the devices' manufacturer. ML algorithms then analyze the collected data compared to the MUD specifications and try to detect any attacks. The considered attacks consist of spoofing, flooding, and DDoS attacks. With a true positive rate of 89.7% at its highest, the authors mention that this might not be enough for large-scale deployments on real IoT networks. This work produced the ACM SOSR 2019 dataset.

In [45], the authors tasked themselves with the creation of an anomaly detection system in the context of enterprise IoT devices. A key point in their contribution is the fact that the data for the ML model may be polluted since the system is located in a real-world environment. The result of their research was RADAR, an ML-based detection system capable of detecting attacks on IoT networks using network traces. The system proved to be robust with an F-Score > 0.9 with up to 15% polluted data [45].

System calls are requests or commands sent from a process or a program to the operating system. Such calls are in most cases regarding access to the memory or file system [7]. The authors of [21] used an interesting and relatively novel approach in using blockchain technology. The system takes a snapshot of the target software and compares it with a trusted and verified snapshot acquired from the blockchain. While acquiring the trusted snapshot, the monitoring system also generates a white list of files that the software can access. The second control mechanism is monitoring file system calls made by the target software and comparing them to its whitelist.

Another system was created by Mishra et al. and called VMGuard [31]. In cloud computing, the authors used Virtual Machine Introspection (VMI) to monitor system call traces made by programs. Using the bag-of-*n*-grams technique, features of normal and attack traces are extracted. Subsequently, an ML classifier, which has been trained on an existing data set [9], is then used to detect any malicious behavior. During their experiments, the authors achieved a detection rate of 94%, up to 100%.

Javaheri et al. proposed a specific framework to detect and neutralize three different types of spyware on Windows [22]. Their approach was to use statistical analysis techniques on kernel-level system calls to classify three types of spyware, keyloggers, screen recorders, and blockers. Their results show that their method had a high accuracy of detecting malware with a 93% detection rate. On the same platform, Canzanese et al. developed another system that uses system call analysis to detect malware [7]. After collecting all the system calls, a bag-of-*n*-grams model was used to extract certain features. The authors showed that they were able to detect unknown malware, similar to a zero-day attack.

In [37] the authors provided an interesting approach where they created MADAM (Multi-Level Anomaly Detector for Android Malware), a system that analyzes the behavior of Android devices on multiple levels. On the kernel level, system calls were

used to see whether the application in question causes any unexpected or irregular increase of activity [37]. Similar to [21], file system calls were used in particular. Using this approach, the authors managed to detect 96% of malicious applications from a dataset containing botnets, ransomware, rootkits, and trojans, among others. In a very similar manner, BRIDEMAID (Behavior-based Rapid Identifier Detector and Eliminator of Malware for AndroID) [28] also uses a multi-level approach considering static and dynamic analysis on the device, user, and application behavior. In general, this work uses a very similar approach as [37] and is also written by the same authors.

Bridges et al. [6] proposed another system to detect malware by analyzing the usage of resources such as the CPU power. The authors defined a series of tasks to be executed on the machine while continuously monitoring the power consumption. Using ML and statistical analysis, the authors differentiated between infected and clean systems.

Barbhuiya et al. proposed the Real-time Anomaly Detection System (RADS) for Cloud Data Centres [4]. RADS monitors and learns the behavior of a virtual machine in a cloud network by analyzing the CPU usage pattern and network traffic usage. In cloud-based computing, there are additional challenges that authors faced while creating the system. One challenge is workload spikes (e.g., during extremely busy hours), resulting in higher CPU usage and potentially leading to false positives. Cloud data centers are especially vulnerable to DDoS and cryptomining attacks [4] which is why RADS is focusing on these attacks. Also, in the context of cloud systems, another interesting approach was taken by the authors of [35]. They proposed a system to detect anomalies by looking at the resource behavior. The motivation behind the idea is that many cloud systems are autoscaling. That means that the system provides any additional required resources. The problem is that a workload created by the malware can request these additional resources. The behavior model consisted of CPU usage metrics used to train an AutoRegressive (AR) model. The model was successfully tested by detecting Denial-of-Service (DoS) and stress attacks.

HPCs are special counters built into computer processors with the task of tracking and counting hardware-related events [38]. Such events can be clock cycles, cache hits/misses, branch behavior, memory access patterns, etc. [40]. This can slightly vary depending on the processor model. Even though HPCs give a deep insight into system performance, only a limited number of them can be tracked simultaneously [5]. The authors of [38] proposed a system, called 2SMaRT, to detect malware using HPCs that consists of two parts. In a first step, 2SMaRT tries to predict if an application is either benignware or part of one of the four defined malware classes, virus, rootkit, backdoor, or trojan. Then, depending on the malware class, a different ML model is deployed to ensure a high detection rate. By using that two-stage model, the authors were able to achieve an F-score of 92% on average.

Similar to [21], the authors of [15] also proposed a detection system using blockchain technology, named CIoTA. Each device of the network trains its detection model locally. Those locally trained models are then merged and validated by the other devices. Opposed to [21], the CIoTA system uses HPCs instead of system calls. The control flow is monitored by creating an Extended Markov Model (EMM)

using HPCs. In a testing environment of 48 Raspberry Pis, CIoTA was able to detect malicious activities successfully.

Interestingly, [46] raised the question of whether malicious behavior actually has an impact on HPC values and, therefore, questioned HPC-based malware detection in general. The authors argue that previous work concerning themselves with HPC-based malware detection presume unrealistic circumstances and assumptions [46]. Mainly the correlation of high-level software behavior, benignware versus malware, and low-level events such as HPCs is questioned. In their experiments, the authors were able to achieve an F1-score of 80.78% at best. Additionally, the authors conducted another experiment where they would induce ransomware into beningware and then see whether ML algorithms could detect said malware using HPCs. No ML model was able to detect the ransomware successfully. Also working on a Windows system, Singh et al. proposed another solution using HPCs in [40]. In their research, the authors identified 16 promising HPCs to detect kernel rootkits specifically using ML algorithms. On a Windows-based testing platform, the system achieved a high detection rate of 99%.

As can be observed, there is a similar distribution of behavior sources as is the case in the previous section regarding datasets. Network is the most used source, but other metrics such as system calls or HPCs are also promising. Additionally, the majority of the reviewed papers used an ML-based approach. It is also worthy to note that many solutions achieved quite a high detection rate for their respective attack type. However, technological advancements and advancements in malware development will further increase the need for newer solutions.

3 Scenario: ElectroSense

To capture and make the radio spectrum data available efficiently, the ElectroSense platform consists of three main components: the sensors, the backend, and the web services (see Fig. 2). The ElectroSense sensors include a single circuit board computer, for example, a Raspberry Pi, a radio frontend, and an antenna, as can be seen in Fig. 1. The sensor used in this work is a Raspberry Pi 4 having an ARMv8 processor with four cores and 4GB of RAM. The antenna collects radio signals processed by the RTL-SDR Silver V3 radio frontend and sent to the server by the Raspberry Pi. The radio frontend monitors the frequency range between 20 MHz and 1.8 GHz. The Raspberry Pi needs to be connected to the internet using either an Ethernet cable or a WiFi dongle. In this work, an Ethernet cable was used. Since the project is an open initiative [33], the software running on the Raspberry Pi is freely available on GitHub.

The ElectroSense backend handles all the data sent by all sensors. Furthermore, it offers an API that allows it to receive raw or aggregated data on every single sensor via an HTTP request. However, certain information like the exact location of a sensor is not available for privacy reasons. The API uses basic authentication, and an ElectroSense account with a username and a password is required to access the API.

Fig. 1 ElectroSense sensor parts

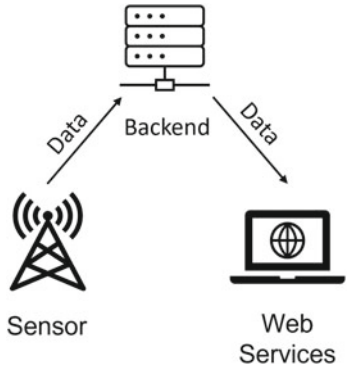


Fig. 2 ElectroSense architecture during the creation of FabIoT

Lastly, the web services offer direct access to the sensors on the ElectroSense website. Each sensor can be accessed, and its live data can be observed through these services. It is also visible where the sensors are located on a map, whether inside or outside and if any other users are accessing the sensor. Privacy settings, such as making the exact location of the sensor accessible to other users, can be changed during the setup process of the sensor. Only one user can access one sensor at a time. While accessing a sensor, the user then chooses between different types of data to view (Table 2).

Table 2 Reviewed attack detection solutions (grouped by behavior source)

Work	Year	Device type	Behavior source	Approach	Tested malware	Performance
[41]	2020	Windows PC	Network/System calls	ML	–	99.54% detection rate
[1]	2020	IoT devices	Network	ML/Blockchain	Botnets [29]	99.2% true positive rate
[16]	2019	IoT devices/SDN	Network	ML	DDoS/Botnets	–
[19]	2019	IoT devices	Network	ML	Network attacks	89.7% True positive rate
[45]	2019	IoT devices	Network	ML	–	F-Score > 0.9
[8]	2018	SDN	Network	ML	DDoS and portscan attacks	Attacks detected and diminished
[21]	2020	IoT devices	Software snap./System calls	Blockchain	Intrusions	Attacks detected
[31]	2020	Cloud systems	System calls	ML	-	94–100% detection rate
[22]	2018	Windows PC	System calls	ML	Spyware/Ransomware	93% detection rate
[37]	2018	Android	System calls	ML	125 Malware families	96% detection rate
[28]	2017	Android	System calls	ML	123 Malware families	99.7% detection rate
[7]	2015	Windows PC	System calls	ML/Statistical	Trojans, viruses and subcategories	92% true positive rate
[6]	2018	Windows PC	Resource usage	ML/Statistical	Rootkits	93–100% detection rate
[4]	2018	Cloud data centres	Resources/Network	ML/Time series	DDoS/Crypotminers	90–95% F1-score
[35]	2018	Cloud systems	Resources usage	Statistical analysis	DoS/Stress attacks	All attacks detected
[38]	2019	Linux PC	HPC	ML	Backdoor, Rootkit, Trojan, Virus	92% F-score on average
[15]	2018	IoT devices	HPC	Blockchain	Exploits	Attacks detected
[46]	2018	Windows PC	HPC	ML	35 different malware families	80.78% F1-score at best
[40]	2017	Windows PC	HPC	ML	Rootkits	99% detection rate

4 Dataset Creation

4.1 Backdoors Affecting ElectroSense

This section illustrates the three chosen backdoor samples. For each one, the focus lies on its features, deployment, and execution. The section closes with a comparison regarding the most important aspects. In a cyber attack setting, the ElectroSense

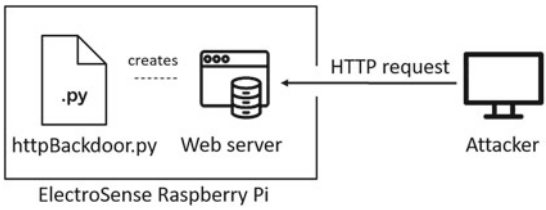


Fig. 3 "httpBackdoor" architecture

Raspberry Pi is considered the client, whereas the system which sends the attacking commands is referenced as the server or attacker side.

4.1.1 httpBackdoor

Features

"httpBackdoor" is arguably the purest form of backdoor of the ones selected in this work. It creates a web server on the Raspberry Pi to which the attacker can send HTTP requests, hence the name "httpBackdoor" [42]. The general architecture of the setup in this work is depicted in Fig. 3. This very simple backdoor has two basic functionalities. Firstly, the attacker can extract basic system information such as OS version and name or SSH keys by sending a GET request. Secondly, by sending a POST request, the attacker can execute command line commands on the Raspberry Pi, as indicated in Fig. 3.

Deployment

The deployment of "httpBackdoor" is rather simple. A Python file needs to be executed on the Raspberry Pi acting as a client. It requires Python to be installed on the Raspberry Pi. However, it is possible to create the binary using *pyinstaller* on another machine and copy it in the Raspberry. Instead of running the Python file on the target machine, which in this setting is the Raspberry Pi, the created binary needs to be executed. Additionally, the binary can be encrypted and thus, making it more difficult to decompile. On the server side, there is no installation required, which is very convenient.

Execution

The attacker needs to include the IP address and port that the Raspberry Pi listens to in the code. To execute the attack, the attacker sends HTTP requests to the specified IP address and port while the Python file or the binary is running on the Raspberry Pi acting as an ElectroSense sensor.

The following commands are examples of how to make use of both functionalities offered by the malware using *curl*. Generally, every tool sending HTTP requests can be used.

- Basic system information extraction
  curl –location –head IP:PORT -H "Content-Type: text"
  This command is used to extract the basic system information.
- Command execution
  curl –location -X POST IP:PORT -H "pwd:password" –data-raw "ls" -i
  This example command sends the "ls" command to the Raspberry Pi and receives information about the current directory. Generally, the format "data-raw" is used. The attentive reader also notices that another header with the content "pwd:password" is also sent as part of the request. "httpBackdoor" offers the possibility for the attacker to set a password, in this case, "password," to ensure that only commands sent by the attacker are executed. Lastly, the additional option "-i" is included so that the response headers which contain the information the attacker is after are displayed.

4.1.2 backdoor

Features

"backdoor" consists of two parts, the client and the server-side [24]. The server-side sends commands to a specified IP address and port on which the malware on the Raspberry Pi listens, as illustrated in Fig. 4. In other words, the malware on the ElectroSense sensor listens for commands from the attacker. The most notable feature is that the backdoor can enable the attacker to open a shell on the Raspberry Pi, giving the attacker complete access. Additionally, the backdoor offers the functionality to pull the contents of a file on the Raspberry Pi onto the attackers' machine.

Deployment

The deployment is very similar to "httpBackdoor" since the client-side malware is written in Python. Generating a binary and then deploying the malware is not explicitly mentioned in the instructions but should also work.

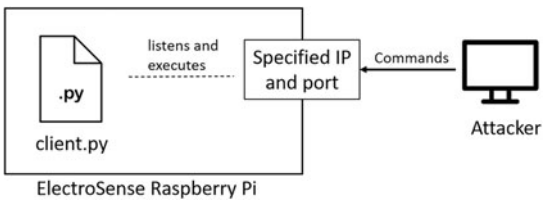


Fig. 4 "backdoor" architecture

Execution

First, it is needed to specify the IP address and port to be included in the server and client Python files. The connection is then established if everything has been set up correctly. If the connection is established successfully, the attacker has the opportunity to execute the following commands on the server-side.

- *shell*. Allows the attacker to open a shell on the Raspberry Pi.
- *recv archive*. Pulls a file from the Raspberry Pi to the attacker machine.
- *help*. Outputs information about the usage of the backdoor.
- *exit*. Terminates the connection and ends the process.

4.1.3 thetick

Features

"thetick" is the most complex of the selected backdoors. Similar to "backdoor", "thetick" resembles a Command and Control (C&C) server structure [32] with the difference that "thetick" can have multiple clients connected at the same time and switch between them seamlessly. In other words, several ElectroSense sensors could potentially be infected and act as a client simultaneously. The attacker can send commands to each client individually, as shown in Fig. 5. In the context of "thetick", the terms, client and bot, are to be understood synonymously.

Deployment

The advanced applicability comes at a price, however. The building and deploying process is more complicated and requires more dependencies than the other two backdoors. On the client side, a Linux executable needs to be built. Depending on the hardware, the process could take some time. The executable can be created

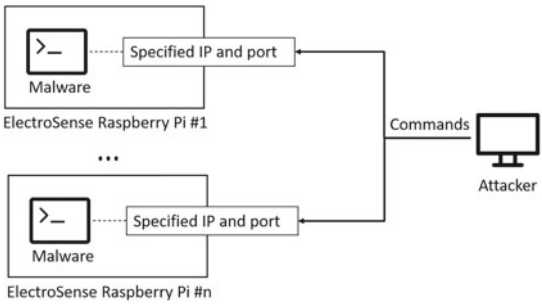


Fig. 5 "thetick" architecture

beforehand on similar hardware and then shipped to the client to avoid it. On the test Raspberry Pi, the build was negligible, however.

Execution

As soon as the executable has been built, it can be executed with the designated IP address, and port passed as arguments. A Python file is executed on the server-side, and the connection should be established automatically. If any other clients are started with the same designated IP address, and port passed as arguments, they also connect automatically, which causes a notification on the server interface. With 18 different commands issued in the C&C server, "thetick" provides extensive possibilities for an attacker to use, which go beyond the traditional definition of a backdoor. The commands have been grouped into either administrative commands or executable commands. The following administrative commands are used to organize different settings on the server-side.

- *bots*. Returns a list of all connected clients at this point of time with the status either "live", "busy" or "gone". Additionally, the IP address, number, and UUID for each client are displayed.
- *clear*. Clears the screen.
- *current*. Shows the currently selected client information.
- *exit*. Exits the program and terminates all connections to clients.
- *use <IP address> <number> <UUID>*. With this command, the attacker can select a new client using one of the three options to identify the bot.

Executable commands are used to interact with the client.

- *chmod <file> <access mode>*. It alters the access mode flag of a file on the Raspberry Pi.
- dig <domain name>. It resolves a domain name at the client.
- *download <url>*. The attacker can download a file onto the Raspberry Pi via HTTP. It can potentially be used to download any further malicious software.
- *exec*. It executes a single non-interactive command (e.g. "ls" or "pwd") and returns the output of the command.
- *fork*. It creates a new instance of a client connection with a new UUID.
- *help * <command>*. It shows additional information in general or per command.
- *kill*. It terminates the connection to the currently selected client and sets its status to "gone".
- *pivot <listen on port> <connect to IP address> <connect to port>*. It is used to create a one-shot TCP tunnel. According to the authors, this is rather useful to launch exploits since the tunnel is only available to localhost and is closed once a client has connected [32].
- *proxy*. Used for proxy management.
  - *proxy [ls]*. Lists all active proxies at the moment.

  - *proxy [add] <port> [bind address] [username] [password]*. Adds a new proxy.
  - *proxy [rm] <port>*. Removes an active proxy.

- *pull <remote file> <local file>*. It pulls the content of a remote file (located on the Raspberry Pi) to a local file.
- *push <local file> <remote file>*. It does the opposite than "pull", the contents of a local file can be pushed into a remote file located on the device.
- *rm <remote file>*. It is used to delete a file on the client.
- *shell*. This command is used to open an interactive shell of the Raspberry Pi and give the attacker full control.

4.1.4 Comparison

Each implementation has its advantages and disadvantages as illustrated in Table 3. "httpBackdoor" is very lightweight, and its code can easily be altered and customized. That means that its limited functionality can be extended in any way one deems fit or necessary. Basic knowledge of Python is required, however. The term "limited" is relative to the other backdoors in this comparison. While not being able to open a shell, "httpBackdoor" still enables the attacker to issue commands and, therefore, have control of the Raspberry Pi. The architecture of "backdoor" is also relatively simple, with the additional requirement of needing Python on the server-side to establish a connection between client and server. The added necessity rewards the attacker with the ability to pull files from the Raspberry Pi. It uses very few dependencies which makes the installation straightforward. Again, basic Python knowledge is required as well. Lastly, "thetick" offers a wide variety of available commands and functionalities which go beyond a regular backdoor. Not only does it provide access to a system, but it also offers options on what to do with that kind of access. It exhibits aspects of a botnet, making it difficult to categorize it into a single malware family. Nevertheless, these similarities are still limited. While it is possible to connect multiple clients to a C&C server and switch between them, it is impossible to issue a simultaneous command to all or multiple clients. There is also one thing that all of the implementations have in common. While they are classified as backdoors, all of them can also act as spyware in their own way. Since they can access the command line in some form, they can provide information about the Raspberry Pi

Table 3 Backdoor comparison

	httpBackdoor	backdoor	thetick
Functionality	Limited	Simple but powerful	Various
Deployment	No installation on server-side	Written in pure Python	Executable on client, Python on server
Architecture	Lightweight	Few dependencies	Complex

to the attacker, which further shows the problem with categorizing malware strictly into one category.

4.2 Malware Attack Behavior

This work defines six different attack behaviors to capture and compare all three backdoors fully. Naturally, because of their differences in complexity, not all backdoors can exhibit all behaviors. In particular, only behavior1–3 can be implemented by all three samples. The behaviors also differ in complexity and, therefore, cause different processes on the Raspberry Pi. Each malware sample achieves the result of each behavior a bit differently. For example, "httpBackdoor" needs to send single commands to perform behavior3, whereas "backdoor" and "thetick" open up a shell on the Raspberry Pi. During the monitoring process, the attacks are repeated every few seconds throughout the specified monitoring time using batch automation scripts. The behaviors are defined as follows:

The goal of **behavior1** is to retrieve some basic system information. For that, simple non-interactive commands such as "pwd" are executed through the respective backdoor. It returns information about where the backdoor execution file is located on the Raspberry Pi. In **behavior2**, a more complex chain of commands is executed. First, a new directory is created and subsequently entered. Next, a repository is cloned into the new directory. In a real-world setting, that could be used to bring further malware onto the device. **behavior3** focuses on deleting files on the target system. To make sure that there are enough files to be deleted, there were roughly 2000 files created before starting the monitoring process. After completing the measurements, the remaining files were deleted before capturing the next behavior. **behavior4** aims to pull a file from the Raspberry Pi to the attacker. In a real-world scenario, that could potentially be used to steal confidential or critical information. In **behavior5**, the goal is to resolve a domain name at the Raspberry Pi. It is especially useful to resolve local domains at the target network. Finally, in **behavior6**, the malware should change access flags for files on the Raspberry Pi. Similar to behavior3, many files were prepared so that the malware would not run out of files to change. Table 4 gives a general overview of the different behaviors.

4.3 Behavioral Monitoring Process

4.3.1 Events

As pointed out in the related work section, system calls, HPCs, and resource usage were among the most promising parameters to model the internal behavior of a resource-constrained device. This work focuses on HPCs and the usage of resources. Unfortunately, the literature fails to mention concrete HPCs for Raspberry Pis. In

Table 4 Attack behaviors

Name	Description
behavior1	Execute a simple non-interactive command ("pwd") and return its ouput
behavior2	Execute a command chain (create a directory and clone a repository)
behavior3	Deleting a file
behavior4	Pulling a file from the Raspberry Pi
behavior5	Resolving a domain name at the Raspberry Pi
behavior6	Changing an access flag for a file on the Raspberry Pi

[38], the authors mention concrete HPCs used to detect backdoors on Linux systems but said HPCs are not supported on the Raspberry Pi test device. Therefore, this work monitors a large amount of different available performance events. The monitoring script collects the values of 72 different performance events during a specified time. Table 5 gives an overview of the selected events and what information they contain.

The authors of [46] argue that detection of malware using HPCs is not possible. However, their arguments assumed that an algorithm could not distinguish between benevolent and malevolent behavior by using HPCs. For example, encrypting a file can be interpreted as a hostile action by ransomware or as a security measure taken by the user. The mentioned problem is the difficulty of linking high-level malware behavior to low-level information such as HPCs [46]. They argue that it is difficult to capture the whole behavior of a device that has such a wide variety of possible and potentially very complex behaviors. A similar observation was made by the authors of [16], where they compared the network behavior of a MacBook Pro and a Philips Hue light bulb. Both are part of an IoT network, but their possible behavior is vastly different. To justify the chosen approach and collect HPCs, we argue that, in our setup, the test device operates inside known parameters, and every deviation from the normal behavior is considered an anomaly and possibly caused by malware.

4.3.2 Monitoring

Architecture

Although the monitoring script is straightforward, there was still a need to organize it in different code blocks. These blocks are described as follows:

- Setup. In this step, any given argument is assigned to a variable. If no argument was given for a variable, the default value is assigned. Additionally, the respective file is created if it does not exist yet. If the file exists, any new monitored data

Table 5 Monitoring events

Events	Information
cpu_usage_pct, ram_usage_pct, network_in, network_out	Usage of resources
block_bio_backmerge, block_bio_remap, block_dirty_buffer, block_getrq, block_touch_buffer, block_unplug	I/O block devices
clk_set_rate	Clock framework
cpu-migrations	CPU migrations
cs	Context switches
fib_table_lookup	Packet forwarding
mm_filemap_add_to_page_cache	File system
gpio_value	GPIO signals
ipi_raise	Inter-Processor interrupts
irq_enable, irq_handler_entry, softirq_entry	Interruption request handling
jbd2_handle_start, jbd2_start_commit	Journaling block device activity
kfree, kmalloc, kmem_cache_alloc, kmem_cache_free, mm_page_alloc, mm_page_alloc_zone_locked, mm_page_free, mm_page_pcpu_drain	Kernel memory
mmc_request_start	Block devices of MMC (Multi media card)
net_dev_queue, net_dev_xmit, netif_rx	Networking
mm_lru_insertion	Kernel interfaces of page tables
qdisc_dequeue	Queuing disciplines
get_random_bytes, mix_pool_bytes_nolock, urandom_read	Kernel random number generator
sys_enter, sys_exit	Quantity of system calls
rpm_resume, rpm_suspend	Runtime power management
sched_process_exec, sched_process_free, sched_process_wait, sched_switch, sched_wakeup	CPU scheduler
signal_deliver, signal_generate	Signals between processes
consume_skb, kfree_skb, skb_copy_datagram_iovec	Socket buffers
inet_sock_set_state	Sockets
task_newtask	Task creation
tcp_destroy_sock, tcp_probe	TCP protocol
hrtimer_start, timer_start	Internal timer
workqueue_activate_work	Work queue
global_dirty_state, sb_clear_inode_writeback, wbc_writepage, writeback_dirty_inode, writeback_dirty_inode_enqueue, writeback_dirty_page, writeback_mark_inode_dirty, writeback_pages_written, writeback_single_inode, writeback_write_inode, writeback_written	System writeback

would simply be appended to the existing file. The performance events that are to be captured are also defined in a string in this section.
- Output Formatting. These lines are responsible for the headers of the data file. The *perf* command needs to be executed because *perf* does not necessarily output the events in the order that were specified earlier in the code. Therefore, the order is extracted from the first execution.
- Monitoring Loop. The monitoring loop consists of two parts, data collection, and output. It runs until the desired amount of samples has been reached. In each loop, the data collection process is executed.
- Data Collection. The main aspect of the monitoring process is being conducted in this section. All the specified data is collected using different commands. The output of the *perf* command is being extracted from a temporary file.
- Output. Lastly, the collected data needs to be added to the final output file as a new sample. If the number of samples has reached its goal, the script terminates at this point.

Execution

A bash shell script is used for the monitoring script because of its ease of use. The script is specifically designed to be executed on a Raspberry Pi, but it is compatible with any Linux OS machine. Two prerequisites need to be installed. However, to capture the performance events, the correct kernel-compatible version of the *perf* tool needs to be installed as well as the *ifstat* tool for collecting network measurements. The output will be a CSV file. The script is executed by issuing the following command:

```
$ bash script.sh [-a] [-f] [-t]
```

To make the script versatile, it has been designed with specific parameters being customizable. That way, the user can easily decide how long the script should run and how many samples should be taken. Finally, several arguments can be passed to customize the output.

- **[-a]** Defines the amount of samples to be taken. The default value is 10.
- **[-f]** Defines the name of the data file. The default value is “data”.
- **[-t]** Defines the time between measurements in seconds. The default is 10.

The list of monitored performance events is also customizable. However, it needs to be altered directly in the code of the script. Another argument that has been implemented is the **[-h]** argument. It is common for most command-line tools to have such a feature to help the user and give an overview of the tool capabilities.

Table 6 Attack behaviors per backdoor type

Backdoor implementation	Considered behaviors
httpBackdoor	behavior1, behavior2, behavior3
backdoor	behavior1, behavior2, behavior3, behavior4
thetick	behavior1, behavior2, behavior3, behavior4, behavior5, behavior6

5 Results

This section discusses the results of the data collection process described in the previous section. For the sake of simplicity, not all results of all 72 events are presented. This work only demonstrates the relevance of the selected events when comparing normal and under attack behavior. Table 6 indicates what behaviors have been considered for which backdoor. For every attack behavior, one file is generated. Each file contains 300 samples with ten-second intervals, and all behaviors are recorded sequentially. After finishing the data collection process for a backdoor sample, the Raspberry Pi is flashed with a new and clean operating system. Thus, traces of the previous backdoor are erased. Each subsection contains one of the malware implementations and analyzes each of the possible behaviors of the respective malware. Additionally, every subsection includes a table containing statistical information like the standard deviation and mean. In the said table, the events are grouped by attack behavior, and for each event, there is statistical information for the clean, labeled as "clean", and infected behavior displayed. A complete version of the dataset is freely available on GitHub [42].

5.1 httpBackdoor

For "httpBackdoor", behavior1–3 were considered. In this section, two events are presented per behavior. Figures 6 and 7 depict behavior1, whereas Figs. 8 and 9 illustrate events under attack behavior2. Finally, Figs. 10 and 11 show events while under attack behavior3.

Figure 6, shows how the RAM usage is significantly higher while the device is experiencing attack behavior1 than during the normal behavior. The numerical data of this event is also presented in Table 7. A possible explanation for that could be the additional resources for running the "httpBackdoor" Python file. On the other hand, some events do not offer such a clear picture. In the case of "httpBackdoor" exhibiting behavior1, an example is presented in Fig. 7.

In contrast to Fig. 7, in Figs. 8 and 9, the events under attack behavior2 are vastly different from the clean behavior. What is most striking is the much greater variation in both events. That is supported by the greater standard deviation visible in Table 7

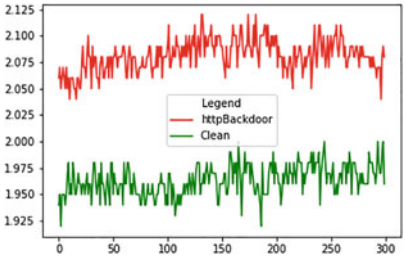


Fig. 6 RAM usage in % for "httpBackdoor" during behavior1

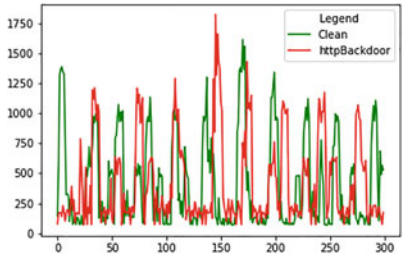


Fig. 7 mm_page_free event for "httpBackdoor" during behavior1

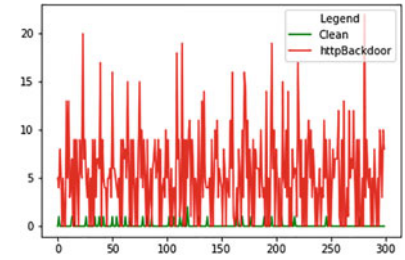


Fig. 8 block_unplug event for "httpBackdoor" during behavior2

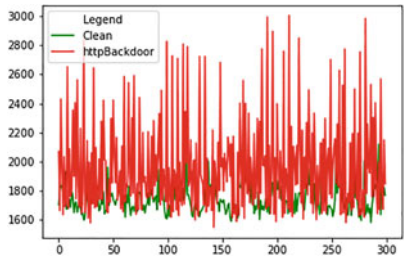


Fig. 9 softirq_entry event for "httpBackdoor" during behavior2

in the behavior2 column. It is logical for behavior2 to cause a lot of activity in the device since it is among the more complex of attack behaviors.

Another clear contrast is presented in Fig. 10. The figure clearly shows the event being different while under attack behavior3. For this event, the standard deviation of the infected behavior is almost eight times as high. Also, the green line is barely visible since there is little activity in the clean behavior as 75% of the values are

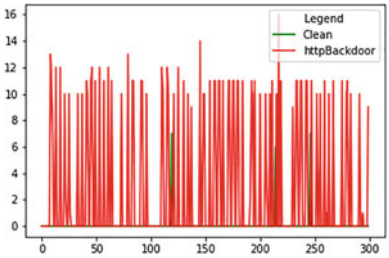


Fig. 10 block_bio_backmerge event for "httpBackdoor" during behavior3

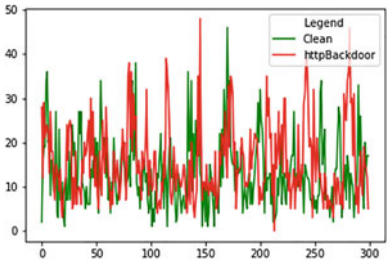


Fig. 11 cpu-migrations event for "httpBackdoor" during behavior3

Table 7 Statistical information of "httpBackdoor"

	behavior1		behavior2		behavior3	
	RAM usage %		block_unplug		block_bio_backmerge	
	clean	http	clean	http	clean	http
mean	1.96	2.08	0.12	5.41	0.07	2.89
std	0.01	0.02	0.35	4.46	0.67	4.79
min	1.92	2.04	0.00	0.00	0.00	0.00
25%	1.95	2.07	0.00	1.00	0.00	0.00
50%	1.96	2.08	0.00	5.00	0.00	0.00
75%	1.97	2.09	0.00	8.25	0.00	9.25
max	2.00	2.12	2.00	22.00	7.00	16.00

equal to 0. Figure 11 does not provide any clear indication of any clear differences between the clean behavior and under attack behavior3.

5.2 backdoor

For "backdoor", behavior1–4 were considered. For each behavior, two events are presented in this section. For behavior1, the qdisc_dequeue event in Fig. 12 and the network_in metric in Fig. 13 are described. Figures 14 and 15 follow and decribe behavior2. Next, behavior3 is illustrated in Figs. 16 and 17. Finally, Figs. 18 and 19 represent behavior.

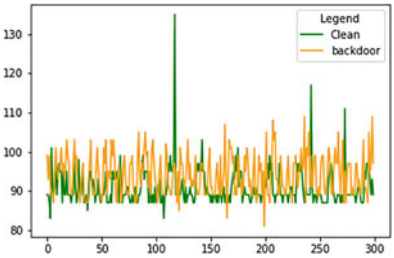


Fig. 12 qdisc_dequeue event for "backdoor" during behavior1

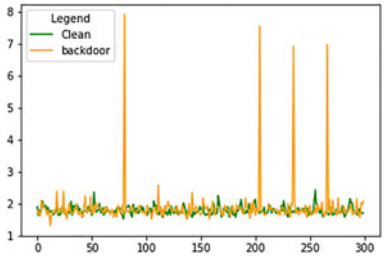


Fig. 13 network_in in kBit/s for "backdoor" during behavior1

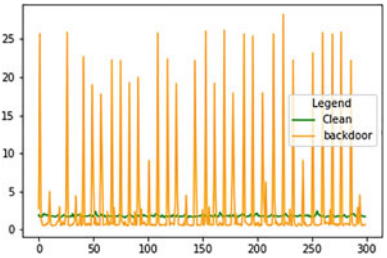


Fig. 14 network_in in kBit/s for "backdoor" during behavior2

There are small but notable differences between the clean and under attack behavior1, as can be observed by investigating Fig. 12. The suspicion is confirmed in Table 8 since both mean and standard deviation are higher during the attack. However, in Fig. 13, these differences are not as clear if outliers are ignored. It seems that in the clean and attack behavior1, the events are fairly consistent and do not indicate any special activity. Since behavior1 is rather simple and does not cause a lot of network traffic, it is logical that the network_in metric is not very different compared to the clean behavior.

However, by looking at Fig. 14, depicting the incoming network traffic during attack behavior2, it is evident that the attack behaviors cause different device behavior. The traffic is substantially higher compared to Fig. 13. That phenomenon can be explained due to the repository cloning that is being caused by attack behavior2. While the clean behavior is very stable and consistent, the infected behavior seems to have higher variation, as confirmed by Table 8. Figure 15 illustrates the mm_page_alloc event, a memory allocation metric. It is clear that during attack behavior2, there is more activity in the memory allocation system.

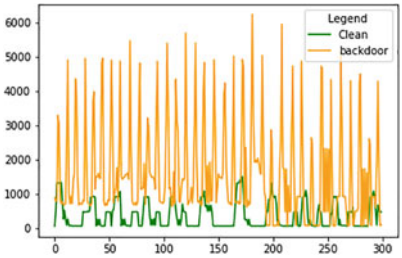


Fig. 15 mm_page_alloc event for "backdoor" during behavior2

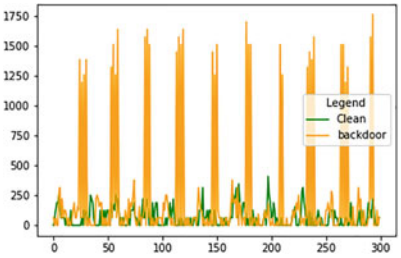


Fig. 16 mm_page_pcpu_drain event for "backdoor" during behavior3

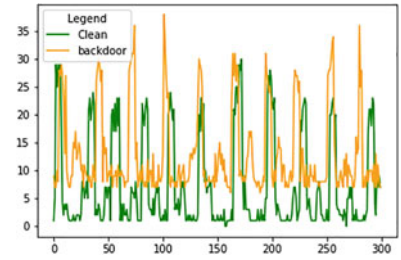


Fig. 17 writeback_dirty_inode event for "backdoor" during behavior3

Two further events that may indicate the presence backdoor under attack behavior3 are illustrated in Figs. 16 and 17. Both events show vast differences compared to normal behavior. While the writeback_dirty_inode event has a similar pattern in both the clear and attack behavior3, the level of the attack data is generally higher. From a statistical view, there are significant differences in both mean and standard deviation for the mm_page_pcpu_drain event, as it is observable in Table 8.

In Fig. 18, the difference between the clean and attack behavior4 is quite large. The "kfree" metric shows the memory that was used and freed. It seems that while the device is under attack, a lot less memory is being freed, possibly because this memory is occupied by the backdoor implementation itself or by the processes the attack behavior has caused. The differences are even clearer in numerical form as can be observed in Table 8 in the behavior4 column. On the other hand, the sys_enter event offers a less clear picture which is further illustrated in Fig. 19.

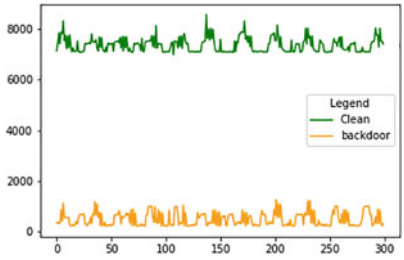


Fig. 18 kfree event for "backdoor" during behavior4

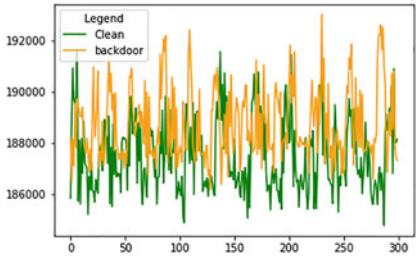


Fig. 19 sys_enter event for "backdoor" during behavior4

Table 8 Statistical information of "backdoor"

	behavior1		behavior2		behavior3		behavior4	
	qdisc_dequeue		network_in		mm_page_pcpu_drain		kfree	
	clean	backdoor	clean	backdoor	clean	backdoor	clean	backdoor
mean	90.80	94.14	1.78	3.16	66.23	218.89	7369.33	491.72
std	4.69	5.16	0.13	6.56	78.94	446.69	280.77	266.24
min	83.00	81.00	1.52	0.43	0.00	0.00	6986.00	213.00
25%	89.00	89.00	1.70	0.54	0.00	0.00	7109.00	247.25
50%	89.00	94.00	1.76	0.66	63.00	63.00	7313.00	370.00
75%	93.00	97.00	1.85	1.02	126.00	133.75	7524.25	677.25
max	135.00	109.00	2.43	28.23	409.00	1764.00	8573.00	1254.00

5.3 thetick

Since "thetick" can exhibit all six different attack behaviors, only one event is presented per behavior. Figure 20 presents the mm_page_alloc_zone_locked during behavior1. For behavior2, Fig. 21 describes the incoming network traffic. In Figs. 22 and 24, the same event describes behavior3 and behavior5 and their differences. For behavior4, the fib_table_lookup event is illustrated in fib_table_lookup Fig. 23. Finally, in Fig. 25 behavior6 is represented.

In Fig. 20, there is another memory allocation related event illustrated under attack behavior1. There are clear variations visible in the figure as well as the data itself in Table 9. Figure 21 displays the same event, also during attack behavior2, as in Fig. 14

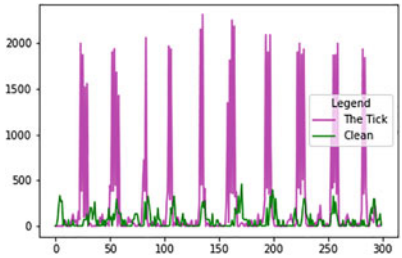


Fig. 20 mm_page_alloc_zone_locked event for "thetick" during behavior1

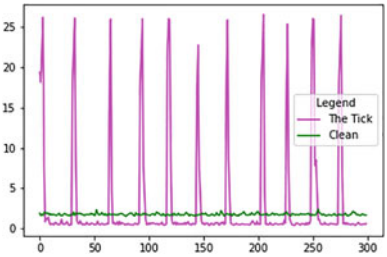


Fig. 21 network_in in kBit/s for "thetick" during behavior2

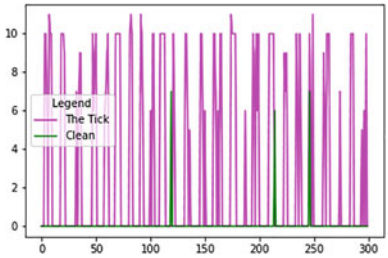


Fig. 22 block_dirty_buffer event for "thetick" during behavior3

with a similar result. It can be observed that although the attacks were conducted using different backdoors, the caused device behavior is very similar. That shows the integrity of the defined attack behaviors across different backdoor implementations.

Another interesting observation is to be made regarding Fig. 22 and 24. Although it is the same performance event, there is no discernible difference between the two behaviors in the latter figure. The explanation for the differences lies in the different attack behaviors. While behavior3 deletes files on the Raspberry Pi, behavior5 resolves domains at the Raspberry Pi and does not interact with the file system. These fundamental differences are clearly visible. It is important to see that certain events can be more or less efficient to detect certain attack behaviors than others. In Fig. 23, similar to Fig. 8, there is a striking contrast in terms of standard deviation between the clean and attack behavior4. Additionally, Table 9 shows significant numerical differences in the aforementioned statistics for the fib_table_lookup event under attack behavior4.

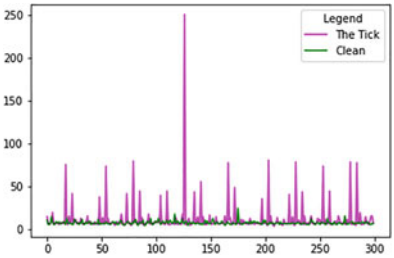


Fig. 23 fib_table_lookup event for "thetick" during behavior4

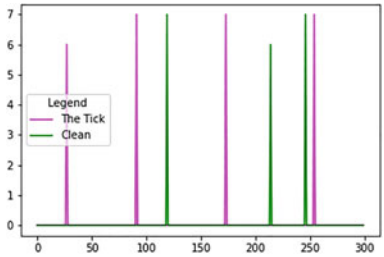


Fig. 24 block_dirty_buffer event for "thetick" during behavior5

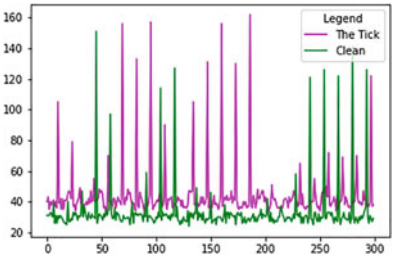


Fig. 25 workqueue_activate_work event for "thetick" during behavior6

The results of the workqueue_activate_work event are presented Fig. 25. It seems that this event might also be a suitable candidate to detect attacks. In the Linux architecture, work queues are a system to manage and distribute the workload to different kernel threads. The workqueue_activate_work event is triggered every time a task is assigned to a thread and subsequently activated. This event seems to have been triggered more during the attacks than during the normal behavior, as can be observed in Table 9. A possible reason for the difference is the additional actions caused by the backdoor during attack behavior6.

Table 9 Statistical information of "thetick"

	behavior1		behavior2		behavior4		behavior6	
	mm_page_alloc_zone_locked		network_in		fib_table_lookup		workqueue_activate_work	
	clean	tick	clean	tick	clean	tick	clean	tick
mean	69.67	253.81	1.78	3.15	7.87	12.79	33.13	44.08
std	92.24	560.89	0.13	6.71	2.07	19.62	16.85	18.56
min	3.00	0.00	1.52	0.43	5.00	4.00	24.00	33.00
25%	4.00	2.00	1.70	0.54	7.00	7.00	28.00	37.00
50%	10.00	33.00	1.76	0.60	7.00	8.00	30.00	40.00
75%	80.25	130.25	1.85	0.80	9.00	10.00	32.00	43.00
max	459.00	2312.00	2.43	26.49	25.00	251.00	151.00	162.00

6 Conclusion

Most of the datasets existing in the literature model the behavior of devices affected by different malware using network-related metrics. However, promising dimensions such as system calls, resource usage, and HPCs exist and are considered more and more. In this context, this chapter proposes FabIoT, a novel dataset modeling the internal behavior of a Raspberry Pi acting as an ElectroSense sensor while being attacked by backdoors with different behaviors. Building upon previous research, the present work shows that detecting backdoors by monitoring device internal behavior is feasible. A script to monitor the internal device behavior while the device is under attack from three real-world backdoors is developed as part of the present work. The backdoors exhibit up to six different attack behaviors to capture a wide variety of independent attacks. The monitoring script outputs a file containing 300 samples of 72 performance events for each possible combination of backdoor and attack behavior. The interval between samples is ten seconds. Then, it was presented that there are significant differences in the collected data and, therefore, illustrated the value FabIoT provides. Preliminary results indicate that different events are more suited to detect attacks than others, depending on the behavior.

Aligned with the current trend, future work in this field contains most likely the creation and development of further datasets. Building upon FabIoT, future datasets could include device fingerprinting while under attack from other types of malware dangerous to IoT networks and platforms, such as botnets and ransomware. Another interesting approach would be to monitor several devices in a real-world IoT platform and conduct attacks on a larger scale compared to the present work, where the focus was on a single device. Moreover, researchers could consider combinations of several malware families as this is often the case in the wild. Another promising avenue to explore is the applicability of HPCs for resource-constrained devices such as the Raspberry Pi used in the present work.

References

1. Ali, J., Khalid, A.S., Yafi, E., Musa, S., Ahmed, W.: Towards a secure behavior modeling for IoT networks using blockchain. arXiv:2001.01841 (2020)
2. Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A., Anwar, A.: Ton IoT telemetry dataset: a new generation dataset of iot and iiot for data-driven intrusion detection systems. IEEE Access **8**, 165130–165150 (2020). https://doi.org/10.1109/ACCESS.2020.3022862
3. Ande, R., Adebisi, B., Hammoudeh, M., Saleem, J.: Internet of things: evolution and technologies from a security perspective. Sustain. Cities Soc. **54**, 101728 (2020). https://doi.org/10.1016/j.scs.2019.101728
4. Barbhuiya, S., Papazachos, Z., Kilpatrick, P., Nikolopoulos, D.S.: RADS: real-time anomaly detection system for cloud data centres. arXiv:1811.04481 (2018)
5. Basu, K., Krishnamurthy, P., Khorrami, F., Karri, R.: A theoretical study of hardware performance counters-based malware detection. IEEE Trans. Inf. Forensic. Secur. **15**, 512–525 (2020). https://doi.org/10.1109/TIFS.2019.2924549
6. Bridges, R., Hernández Jiménez, J., Nichols, J., Goseva-Popstojanova, K., Prowell, S.: Towards malware detection via cpu power consumption: data collection design and analytics. In: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), pp. 1680–1684 (2018). https://doi.org/10.1109/TrustCom/BigDataSE.2018.00250
7. Canzanese, R., Mancoridis, S., Kam, M.: System call-based detection of malicious processes. In: 2015 IEEE International Conference on Software Quality, Reliability and Security, pp. 119–124 (2015). https://doi.org/10.1109/QRS.2015.26
8. Carvalho, L.F., Abrão, T., Mendes, L.D.S., Proença, M.L.: An ecosystem for anomaly detection and mitigation in software-defined networking. Exp. Syst. Appl. **104**, 121–133 (2018). ISSN 0957-4174. https://doi.org/10.1016/j.eswa.2018.03.027
9. U.o.N.M. Computer Science Department. Computer immune systems (2021). https://www.cs.unm.edu/immsec/systemcalls.htm. Accessed 15 June 2021
10. Creech, G.: Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks. Ph.D. thesis, University of New South Wales (2014)
11. Creech, G., Hu, J.: A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns. IEEE Trans. Comp. **63**(4), 807–819 (2013). https://doi.org/10.1109/TC.2013.13
12. Damasevicius, R., Venckauskas, A., Grigaliunas, S., Toldinas, J., Morkevicius, N., Aleliunas, T., Smuikys, P.: Litnet-2020: an annotated real-world network flow dataset for network intrusion detection. Electronics **9**(5) (2020). https://doi.org/10.3390/electronics9050800
13. Gandotra, E., Bansal, D., Sofat, S.: Zero-day malware detection. In: 2016 Sixth International Symposium on Embedded Computing and System Design (ISED)
14. Garcia, S., Parmisano, A., Erquiaga, M.J.: IoT-23: a labeled dataset with malicious and benign IoT network traffic (2020). https://www.stratosphereips.org/datasets-iot23. Accessed on 22 Oct 2021
15. Golomb, T., Mirsky, Y., Elovici, Y.: CIOTA: collaborative IoT anomaly detection via blockchain (2018). arXiv: 1803.03807 [cs.CY]
16. Haefner, K., Ray, I.: Complexiot: behavior-based trust for IoT networks. In: 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), pp. 56–65 (2019). https://doi.org/10.1109/TPS-ISA48467.2019.00016
17. Hafeez, I., Antikainen, M., Ding, A.Y., Tarkoma, S.: IoT-keeper: detecting malicious IoT network activity using online traffic analysis at the edge. IEEE Trans. Netw. Serv. Manage. **17**(1), 45–59 (2020). https://doi.org/10.1109/TNSM.2020.2966951
18. Haider, W., Hu, J., Slay, J., Turnbull, B., Xie, Y.: Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. J. Netw. Comput. Appl. **87**, 185–192 (2017). ISSN 1084-8045. https://doi.org/10.1016/j.jnca.2017.03.018

19. Hamza, A., Gharakheili, H.H., Benson, T.A., Sivaraman, V.: Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity. In: 2019 ACM Symposium on SDN Research, SOSR'19, pp. 36–48, San Jose, CA, USA (2019). ISBN 9781450367103. https://doi.org/10.1145/3314148.3314352
20. Hashemi, S., Zarei, M.: Internet of things backdoors: resource management issues, security challenges, and detection methods. Trans. Emerg. Telecommun. Technol. **32**(2), e4142 (2021). https://doi.org/10.1002/ett.4142
21. He, S., Ren, W., Zhu, T., Choo, K.-K.R.: Bosmos: a blockchain-based status monitoring system for defending against unauthorized software updating in industrial internet of things. IEEE Internet Things J. **7**(2), 948–959 (2020). https://doi.org/10.1109/JIOT.2019.2947339
22. Javaheri, D., Hosseinzadeh, M., Rahmani, A.M.: Detection and elimination of spyware and ransomware by intercepting kernel-level system routines. IEEE Access **6**(78321–78332), 2884964 (2018). https://doi.org/10.1109/ACCESS.2018
23. Kang, H., Ahn, D.H., Lee, G.M., Yoo, J.D., Park, K.H., Kim, H.K.: IoT network intrusion dataset. 2019. https://dx.doi.org/10.21227/q70p-q449. Accessed on 22 Oct 2021
24. Koritar, J.: Backdoor (2020). https://github.com/jakoritarleite/backdoor Accessed on 22 Oct 2021
25. Lashkari, A.H., Kadir, A.F.A., Gonzalez, H., Mbah, K.F., Ghorbani, A.A.: Towards a network-based framework for android malware detection and characterization. In: 2017 15th Annual Conference on Privacy, Security and Trust (PST), pp. 233–233 (2017). https://doi.org/10.1109/PST.2017.00035
26. Lashkari, A.H., Kadir, A.F.A., Taheri, L., Ghorbani, A.A.: Toward developing a systematic approach to generate benchmark android malware datasets and classification. In: 2018 International Carnahan Conference on Security Technology (ICCST)
27. Lu, Y., Xu, L.D.: Internet of things (IoT) cybersecurity research: a review of current research topics. IEEE Internet Things J. **6**(2), 2103–2115 (2019). https://doi.org/10.1109/JIOT.2018.2869847
28. Martinelli, F., Mercaldo, F., Saracino, A.: Bridemaid: an hybrid tool for accurate detection of android malware. In: 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS'17, Abu Dhabi, United Arab Emirates, pp. 899– 901 (2017). ISBN 9781450349444. https://doi.org/10.1145/3052973.3055156
29. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., Elovici, Y.: N-BAIoT-network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Comput. **17**(3), 12–22 (2018). https://doi.org/10.1109/MPRV.2018.03367731
30. Or-Meir, O., Nissim, N., Elovici, Y., Rokach, L.: Dynamic malware analysis in the modern era-a state of the art survey. ACM Comput. Surv. **52**(5) (2019). ISSN 0360-0300. https://doi.org/10.1145/3329786
31. Mishra, P., Varadharajan, V., Pilli, E.S., Tupakula, U.: Vmguard: a VMI-based security architecture for intrusion detection in cloud environment. IEEE Trans. Cloud Comput. **8**(3), 957–971 (2020). https://doi.org/10.1109/TCC.2018.2829202
32. nccgroup. The tick (2020) https://github.com/nccgroup/thetick. Accessed on 22 Oct 2021
33. Network, T.E.: Electrosense—collaborative spectrum monitoring (2021). https://electrosense.org. Accessed on 01 July 2021
34. Pahl, M.-O., Aubet, F.-X.: All eyes on you: distributed multi-dimensional IoT microservice anomaly detection. In: 2018 14th International Conference on Network and Service Management (CNSM), pp. 72–80 (2018)
35. Ravichandiran, R., Bannazadeh, H., Leon-Garcia, A.: Anomaly detection using resource behaviour analysis for autoscaling systems. In: 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), pp. 192–196 (2018). https://doi.org/10.1109/NETSOFT.2018.8460025
36. Sánchez, P.M., Jorquera Valero, J.M., Huertas Celdrán, A., Bovet, G., Gil Pérez, M., Martínez Pérez, G.: A survey on device behavior fingerprinting: data sources, techniques, application scenarios, and datasets. IEEE Commun. Surv. Tutor. **23**(2), 1048–1077 (2021). https://doi.org/10.1109/COMST.2021.3064259

37. Saracino, A., Sgandurra, D., Dini, G., Martinelli, F.: Madam: effective and efficient behavior-based android malware detection and prevention. IEEE Trans. Dependable Secure Comput. **15**(1), 83–97 (2018). https://doi.org/10.1109/TDSC.2016.2536605
38. Sayadi, H., Makrani, H.M., Pudukotai Dinakarrao, S.M., Mohsenin, T., Sasan, A., Rafatirad, S., Homayoun, H.: 2smart: a two-stage machine learning-based approach for run-time specialized hardware-assisted malware detection, pp. 728–733 (2019). https://doi.org/10.23919/DATE.2019.8715080
39. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
40. Singh, B., Evtyushkin, D., Elwell, J., Riley, R., Cervesato, I.: On the detection of kernel-level rootkits using hardware performance counters. In: 2017 ACM on Asia Conference on Computer and Communications Security, pp. 483–493 (2017). ISBN 9781450349444. https://doi.org/10.1145/3052973.3052999
41. Singh, J., Singh, J.: Detection of malicious software by analyzing the behavioral artifacts using machine learning algorithms. Inf. Softw. Technol. **121**, 106273 (2020). ISSN 0950-5849. https://doi.org/10.1016/j.infsof.2020.106273
42. Sisi, F.: Fabiot-dataset (2021). https://github.com/Fabiooo98/FabIoT-Dataset Accessed on 22 Oct 2021
43. SkryptKiddie. httpBackdoor (2020). https://github.com/Skrypt-Kiddie/httpBackdoor. Accessed on 22 Oct 2021
44. Taheri, L., Kadir, A.F.A., Lashkari, A.H.: Extensible android malware detection and family classification using network-flows and api-calls. In: 2019 International Carnahan Conference on Security Technology (ICCST), pp. 1–8 (2019). https://doi.org/10.1109/CCST.2019.8888430
45. Yu, T., Sun, Y., Nanda, S., Sekar, V., Seshan, S.: RADAR: A robust behavioral anomaly detection for IoT devices in enterprise networks. Technical Report CMU-CyLab-19-003, Carnegie Mellon University (2019)
46. Zhou, B., Gupta, A., Jahanshahi, R., Egele, M., Joshi, A.: Hardware performance counters can detect malware: myth or fact? In: 2018 on Asia Conference on Computer and Communications Security, ASIACCS'18, Incheon, Republic of Korea, pp. 457–468 (2018). ISBN 9781450355766. https://doi.org/10.1145/3196494.3196515

Anomaly Detection on Static and Dynamic Graphs Using Graph Convolutional Neural Networks

Amani Abou Rida, Rabih Amhaz, and Pierre Parrend

Abstract Anomalies represent rare observations that vary significantly from others. Anomaly detection intended to discover these rare observations and has the power to prevent detrimental events, such as financial fraud, network intrusion, and social spam. However, conventional anomaly detection methods cannot handle this problem well because of the complexity of graph data (e.g., irregular structures, relational dependencies, node/edge types/attributes/directions/multiplicities/weights, large scale, etc.) (Ma X, Wu J, Xue S, Yang J, Zhou C, Sheng QZ, Xiong H, Akoglu L. IEEE Trans Knowl Data Eng, 2021 [1]). Thanks to the rise of deep learning in solving these limitations, graph anomaly detection with deep learning has obtained an increasing attention from many scientists recently. However, while deep learning can capture unseen patterns of multi-dimensional Euclidean data, there is a huge number of applications where data are represented in the form of graphs. Graphs have been used to represent the structural relational information, which raises the graph anomaly detection problem—identifying anomalous graph objects (i.e., vertex, edges, sub-graphs, and change detection). These graphs can be constructed as a static graph, or a dynamic graph based on the availability of timestamp. Recent years have observed a huge efforts on static graphs, among which Graph Convolutional Network

These authors contributed equally to this work.

A. A. Rida (✉) · R. Amhaz · P. Parrend
ICube (Laboratoire des sciences de l'ingénieur, de l'informatique et de l'imagerie), UMR 7357, Université de Strasbourg, CNRS, 67000 Strasbourg, France
e-mail: abou-rida.amani@etu.unistra.fr

R. Amhaz
e-mail: amhaz@unistra.fr

P. Parrend
e-mail: pierre.parrend@epita.fr

R. Amhaz
ECAM Strasbourg-Europe, 67300 Schiltigheim, France

P. Parrend
EPITA, 5, Rue Gustave Adolphe Hirn, 67000 Strasbourg, France

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_12

(GCN) has appeared as a useful class of models. A challenge today is to detect anomalies with dynamic structures. In this chapter, we aim at providing methods used for detecting anomalies in static and dynamic graphs using graph analysis, graph embedding, and graph convolutional neural networks. For static graphs we categorize these methods according to plain and attribute static graphs. For dynamic graphs we categorize existing methods according to the type of anomalies that they can detect. Moreover, we focus on the challenges in this research area and discuss the strengths and weaknesses of various methods in each category. Finally, we provide open challenges for graph anomaly detection using graph convolutional neural networks on dynamic graphs.

Keywords Anomaly detection · Graph anomaly detection · Graph analysis · Graph embedding · Graph neural network · Dynamic graphs · Static graphs

1 Introduction

Anomaly detection is a critical task that deals with the problem of discovering "different from normal" signals or patterns by evaluating a large amount of data, thereby major flaws can be identified and avoided [2]. The objective of anomaly detection is to identify the data that are significantly different from most observations. Research on anomaly detection dates all the way back to the 1980s and detecting anomalies on graphs using machine learning has become an important research problem [2]. Graph anomaly detection has evolved from depending heavily on human experts' domain information into machine learning methods to eliminate human intervention. Recently, different deep learning approaches have been taken on to distinguish potential anomalies in graphs. These approaches are graph analysis [3], graph embedding [3], and graph neural networks [4]. By leveraging the graph structure, several of novel algorithms for anomaly detection have been proposed for each approach.

In recent years, graph neural networks (GNNs), as a powerful deeplearning-based graph representation method, have exhibited superiority in leveraging the graph structure and has been utilized in anomaly detection. Some researchers have successfully applied GNNs in several important anomaly detection tasks. Based on the availability of the timestamp, the graph can be constructed as a static graph or a dynamic graph, where a static graph refers to the graph that has fixed nodes and edges, and a dynamic graph refers to the graph that has nodes and/or edges change over time [2]. Dynamic graphs are more complex and difficult in discovering anomalies than static graphs. This can be shown from two perspectives: (1) The anomalous edges cannot be determined by the graph from a single timestamp. The detection procedure must take graph history into consideration; (2) Both the vertex and edge sets are changing over time [5]. As graph-based anomaly detection is becoming ever more important and the achievements of graph neural networks are increasing, both academia and industry are interested in applying GNNs to handle the issue of anomaly detection on static and dynamic graphs. In this chapter, we summarize different GNN-based

anomaly detection methods and provide taxonomies for them according to static and dynamic graphs. We will also see how GNN methods solved the limitations of graph analysis and graph embedding on both static and dynamic graphs.

The rest of this chapter is organized as follows. Section 2 identifies the approaches for analyzing graphs. Section 3 identifies the key challenges of Graph Neural Networks, and for Anomaly Detection on dynamic graphs. Section 4 presents methods for anomaly detection on static graphs and Sect. 5 presents methods for anomaly detection on dynamic graphs. Section 6 identifies the requirements needed for better anomaly detection. In the last section, we provide the conclusion of this work.

2 Analyzing Graphs

In this section we define the three core methods for analyzing graphs: graph analysis, graph embedding, and graph neural networks.

2.1 Graph Analysis

There is an increasing number of applications where data are represented as a graph with complex relationships and inter-dependency between objects. This means going from Euclidean (e.g., images, audio, and text) to non-Euclidean space representing interactions between nodes instead of the characteristics of individual points. Graph analysis [3] is a process for analyzing data in graph structures, in a **Non-Euclidean Space** using data points as nodes and relationships as edges. To perform the analysis on a non-Euclidean space, graph analysis methods have come into to improve the quantitative understanding and the control of complex networks.

2.2 Graph Embedding

Graph Embedding [3] methods have shown an important role for the capacity of transforming high-dimensional sparse graphs into **low-dimensional representations**, dense and continuous vector spaces. The main aim of graph embedding techniques is to encode nodes into a latent vector space and to group every node's property into a vector with a lower dimension.

2.3 Graph Neural Networks

Graph neural networks (GNNs) are classified into 4 categories: recurrent graph neural networks (RecGNNs), convolutional graph neural networks (ConvGNNs), graph autoencoders (GAEs), and spatial–temporal graph neural networks (STGNNs) [4]. In this chapter we will focus on Graph Convolutional Neural Networks (ConvGNNs) that acquire the movement of convolution from grid data (Euclidean structure) to graph data (non-Euclidean structure) [4].

ConvGNNs play an important role in building up many other complex GNN models [4]. They fall into two categories, spectral-based and spatial-based GNNs. Spectral based methods determine graph convolutions by proposing filters from graph signal processing [6]. Spatial-based methods achieve graph convolutions locally on each node where weights can be easily shared across different locations and structures [7].

Most of the current graph neural network methods have been modeled for static graphs, while many real-world graphs are being dynamic. Concurrently, designing graph neural networks for dynamic graphs is facing challenges. From the global perspective, structures of dynamic graphs remain evolving since new nodes and edges are always introduced. It is necessary to track the changing of graph neural network's structure. From the local perspective, a node can carry new edges, and the order of these edges is critical for learning the node [8]. In the next section we will identify the challenges for graph neural networks.

3 Key Challenges

In this section we will first present the key challenges for graph neural networks and then identify the challenges for anomaly detection on dynamic graphs.

3.1 Challenges for Graph Neural Networks

While Graph Neural Networks have proved to be a very powerful approach for learning graph data, there are still several open challenges due to the complexity of graphs. Some of the challenges are listed below:

- **Model Depth**: [4] Deep learning model achievements depend on the building of neural networks. However, when using graphs, experimental studies have shown that with the increase in the number of layers, the model performance we be dropped dramatically [9]. This is caused by the effect of graph convolutions in pushing representations of adjacent nodes closer to each other. So it is noticed that with infinite number of convolutions, all nodes' representations will move

to a single point. So, the challenge here is to evaluate whether going deep in the neural network layers can still improve graph data learning.

- **Scalability**: [4] At any time we try to scale or cluster our graph, it is an irrelevant problem that the completeness of the graph is sacrificed. The model will lose some part of graph information whether we are scaling or clustering. For scaling, a node can drop its significant neighbors. For clustering, a graph can drop a different structural pattern. Here comes the challenge of how it is possible to handle the scalability without sacrificing the integrity.
- **Heterogeneity**: [10] It is widely seen that most of GNN methods are applied on homogeneous graphs. The current GNN methods cannot handle the heterogeneous graph data, as it may have various types of nodes and edges, or it may have different features. This calls for new approaches that can handle such heterogeneous data.
- **Dynamicity**: [10] Graphs have dynamic nature in a way that the nodes and edges keep changing. They can appear at some time and disappear at some other. Dynamic graphs are more complex due to the change of the graph structure. That is, the vertices and edges are unstable along the time dimension. Sometimes the nodes variate according to the time and the environment.

A challenge today is to deal with graphs with dynamic structures. It is important to study the anomaly detection on dynamic graphs using GNNs. The real-world networks can be modeled as dynamic graphs to represent the evolving objects and relationships among them. Apart from the structural information and node attributes, dynamic graphs also contain affluent temporal signals. On the one hand, this information inherently makes anomalous node detection on dynamic graphs more challenging. This is because dynamic graphs normally introduce massive volume of data and temporal signals must also be captured. On the other hand, they could provide more details about anomalies [8, 11, 12]. For instance, some anomalous nodes can appear to be normal in the graph snapshot at each time stamp, only when the graph structure variations are considered, they can be detected. In the next section we will provide the challenges for anomaly detection on dynamic graphs [1].

3.2 Challenges for Anomaly Detection on Dynamic Graphs

Although anomaly detection has been an important research area for several years, there are still some unique and complex nature of anomalies which leads to unsolved challenges. Anomaly detection is not trivial due to its flexible and dynamic nature. Some anomalous operations display few explicit patterns but try to hide them in a large graph, while other operations are with constant patterns [13]. The core challenges for anomaly detection in dynamic graphs are the following ones:

- Some graphs are evolving over time which leads to new types of anomalies, for example, splitting, disappearing, or flickering communities [8].

- Graphs from many domains can exhibit entirely different behaviors over time. This divergence in evolution leads to application and approaches for specific anomalies [8].
- Some anomalies are slow to develop and span multiple time steps, thus it can be difficult to differentiate from organic graph evolution [8].
- The relationships between real objects and their inter-dependencies can no longer be treated individually for anomaly detection. The detection methods need to digest the deviating patterns of anomalies by considering the pairwise, triadic, and higher relationships among objects restored in conventional graphs [14, 15]. Moreover, the dynamic nature of real networks makes the detection problem much more challenging when the graph structure and attributes of nodes or edges change overtime [1].

4 Anomaly Detection on Static Graphs

In this section we define two types of static graphs: plain static, and attribute static graphs. We also define the methods used and the limitations of the three approaches: graph analysis, graph embedding, and graph convolutional neural networks in plain and attribute static graphs. Figure 1 shows how the methods are categorized.

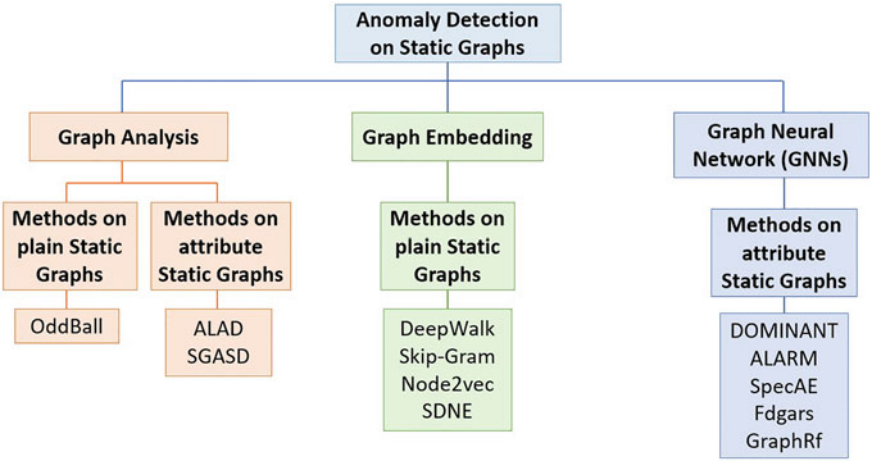


Fig. 1 Anomaly detection methods applied on static graphs

4.1 Definitions

Plain Static Graph

A static plain graph $G = \{V, E\}$ contains a node set V and an edge set E. In a static plain graph, the graph structure consists of nodes $V = \{v_i\}$ and edges $E = \{e_{i,j}\}$ where n expresses the number of nodes and $e_{i,j} = (v_i, v_j)$ denotes an edge between nodes $v_i$and v_j. The $n \times n$ adjacency matrix $A = [a_{i,j}]$ restores the graph structure, where $a_{i,j} = 1$ if there exists an edge between nodes $v_i$and v_j, else $a_{i,j} = 0$ [1].

Attribute Static Graph

A static attributed graph $G = \{V, E, X\}$ contains a node set V, an edge set E and an attribute set X. In an attributed graph, the graph structure follows the previous definition. The $n \times k$ attribute matrix $X = [x_i]$ consists of nodes' attribute vectors, where x_i is the attribute vector associated with node v_i and k is the vector's dimension [1].

4.2 Anomaly Detection on Static Graphs Using Graph Analysis

Existing graph analysis methods for anomaly detection are established from machine learning models. The supervised learning methods learn the patterns of abnormal events and fit the non-linear models. These methods are reasonable for labeling the data as anomalies. The semi-supervised learning methods apply less labeled data to meet the models and detect the abnormal events on the unlabeled data. In general, these methods are constructed through an auto-encoder. If the loss induced by a data point is greater than the threshold, the data point is detected as an anomaly [16]. Normally these methods are not capable of detecting the anomaly without untrained patterns [16]. In this section we will divide the methods for anomaly detection using graph analysis into plain graphs and attribute graphs as follows.

4.2.1 Methods for Plain Static Graphs

Plain graphs are dedicated to representing the structural information in realworld networks. For anomaly detection in plain graphs, the graph structure has been broadly exploited from different angles. A main idea behind that is to change the graph anomaly detection into a traditional anomaly detection issue, because the graph data with strong structure information cannot be handled by traditional detection methods. To bridge the gap, lots of works [17–19] manage to leverage the statistical features associated with each node such as in/out degree to detect anomalous nodes. For instance, OddBall [17] employs the statistical features (e.g., the number of 1-hop neighbors and edges, the total weight of edges) extracted from each node and its

1-hop neighbors to detect special structural anomalies that: (1) form local structures in shape of near-cliques or stars, (2) have heavy links with neighbors such that the total weight is extremely large, or (3) have a single dominant heavy link with one of the neighbors [1].

4.2.2 Limitations

When dealing with real word scenarios, it is very hard to choose the most suitable features from many candidates, and domain experts can always design new statistics, e.g., the maximum/minimum weight of edges. As a result, these methods lead to excessive costs for assessing the most significant features and cannot capture the structural information completely [1].

4.2.3 Methods for Attribute Static Graphs

Apart from the structural information, real-world networks also contain affluent attribute information affiliated with nodes [20, 21]. These attributes supply complementary information over real objects and together with graph structure, more unseen anomalies that are non-trivial can now be discovered. Traditional techniques (e.g., statistical models, matrix factorization, KNN) have been widely applied to extract the structural/attribute patterns of anomalous nodes, after which the detection is performed. Among them, matrix factorization (MF) based techniques have shown power on capturing both the topological structure and node attributes. They achieve promising detection performance. For example, ALAD [22] measures the normality of each node according to attribute similarity with the community it belongs to. By ranking the nodes' normality scores in ascending order, the top-k nodes are identified as community anomalies. Linear regression models are also adopted to train anomaly classifiers given the labeled training data [1]. A representative work in [23] which is a supervised model, SGASD, that has yield encouraging results on identifying social spammers utilizing the social network structure, content information in social media and user labels [1].

4.2.4 Limitations

These graph analysis methods capture valuable information from the graph topology and node attributes, but their applications and generalizability's to real networks in large-scale are strictly limited due to the high computational cost of the matrix decomposition operation and regression models [1].

4.3 Anomaly Detection on Static Graphs Using Graph Embedding

In this section we present an overview of graph embedding methods that are used for anomaly detection on plain graphs.

4.3.1 Methods for Plain Static Graphs

Graph embedding methods have been widely used to capture more valuable information from the graph structure for anomaly detection. Typically, these methods encode the graph structure into an embedded vector space and identify anomalous nodes through further analysis [1]. The first method of graph embedding is DeepWalk [11]. DeepWalk transforms graph structure information into sequences by random walk, and then Skip-Gram [24] is applied on it for each node embedding. After the success of DeepWalk, more graph embedding methods were designed. Node2vec [25] improves the random walk strategies of DeepWalk by a controllable deep or wide walking possibility. SDNE [26] introduces the Deep Auto-encoder model into graph embedding, modeling the 1st-order and 2ND-order neighbors of nodes. Using the graph embedding methods Skip gram, Node2vec, Deepwalk, both anomalous node and edge detection tasks can be applied with standard anomaly detection methods.

4.3.2 Limitations

Researchers easily use static graph embedding methods in every snapshot during different time steps, with constraint to align the same nodes in discrete time steps. In this way, researchers claim that dynamic graph embedding methods could obtain better representations than traditional static graph embedding, since different snapshots share more characteristics for representation.

4.4 Anomaly Detection on Static Graphs Using Graph Convolutional Neural Network

In this section we present an overview of graph convolutional neural network methods that are used for anomaly detection on attribute graphs.

4.4.1 Methods for Attribute Static Graphs

Graph convolutional neural networks (ConvGNNs) [27] have accomplished decent success in many graph data mining tasks (e.g., link prediction, node classification, and

recommendation) owing to its capability of capturing comprehensive information in the graph structure and node attributes. Therefore, many anomalous node detection techniques start to investigate ConvGNNs. The proposed method, DOMINANT [28], comprises three parts, namely, graph convolutional encoder, structure reconstruction decoder, and attribute reconstruction decoder. The graph convolutional encoder generates node embeddings through multiple graph convolutional layers. For the purpose of capturing these signals, the proposed method, ALARM [29], applies multiple ConvGNNs to encode information in different views and adopts a weighted aggregation of them to generate node representations. The training strategy of this model is similar to DOMINANT [28] and aims at minimizing network reconstruction loss and attribute reconstruction loss. Lastly, ALARM adopts the same scoring function as [29], and nodes with top-k highest scores are anomalous. Instead of spotting unexpected nodes using their reconstruction errors, SpecAE [30] is used to detect global anomalies and community anomalies via a density estimation approach, Gaussian Mixture Model (GMM). The global anomalies can be identified by only considering the node attributes. For community anomalies, because of their distinctive attributes to the neighbors, the structure and attributes need to be jointly considered. Accordingly, SpecAE investigates a graph convolutional encoder to learn node representations and reconstruct the nodal attributes through a deconvolution decoder. In Fdgars [31] they developed a novel detection model that can identify fraudsters using their relations and features. This method firstly models online users' reviews and visited items as their features, and then identifies a small portion of significant fraudsters based on these features. In the last step, a ConvGNNs is trained in a semi-supervised manner by using the user network, user features, and labeled users. After training, the model can label unseen users directly. A more recent work, GraphRfi [32], also explores the potential of combining anomaly detection with other downstream graph analysis tasks. It targets on leveraging anomaly detection to identify malicious users and to provide more accurate recommendations to service benign users by alleviating the impact of these untrustworthy users.

4.4.2 Limitations

ConvGNNs has a simple convolution operation that aggregates neighbour information equally to the target node. This helps ConvGNNs to provide an effective solution to incorporate the graph structure with node attribute. However, ConvGNNs capability in capturing the most relevant information from neighbors is unsatisfactory.

5 Anomaly Detection on Dynamic Graphs

The real-world networks can be modeled as dynamic graphs to represent the evolving objects and relationships among them. In this section we define dynamic graph and

present the type of anomalies in dynamic graph. We also present the methods used and the limitations of the three approaches: graph analysis, graph embedding, and graph convolutional neural network according to the different types of anomalies in dynamic graph.

5.1 Definitions

Dynamic Graphs

In dynamic graphs, vertices and edges can be inserted or removed at every time step. For simplicity, we assume that the vertex correspondence and the edge correspondence across different time steps are resolved because of unique labeling of vertices and edges, respectively. We define a graph series G as an ordered set of graphs with a fixed number of time steps. Formally, $Gt_t = 1^T$ where T is the total number of time steps, $G_t = (V_t, E_t \subseteq (V_t \times V_t))$, and the vertex set V_t and edge set E_t may be plain or attributed (labeled). Graph series where $T \to \infty$ are called graph streams [1].

Types of anomalies on dynamic graphs

In this section, we identify and formalize four types of anomalies that arise in dynamic graphs.

1. **Anomalous Vertices** Anomalous vertex detection intends to find a subset of the vertices such that every vertex in the subset has an 'irregular' evolution compared to the other vertices in the graph [8]. Alternatively, the time points where the vertices are supposed to be anomalous can be detected. Dynamic graphs allow the temporal dynamics of the vertex to be involved, offering new types of anomalies that are not showed in static graphs. Anomalous vertices is formally defined as follows:

   Definition 1: Anomalous vertices (from [8]) Given G, the total vertex set $V = \bigcup_{t=1}^{T} V_t$, and a specified scoring function $f : V \to \mathbb{R}$, the set of anomalous vertices $V' \subseteq V$ is a vertex set such that $\forall\, v' \in V',\ |f(v') - \hat{f}| > c_0$, where $\hat{f}$ is a summary statistic of the scores $f(v), \forall\, v \in V$.
2. **Anomalous Edges** Similar to vertex detection, edge detection aims to find a subset of the edges such that every edge in the subset has an 'irregular' evolution, optionally identifying the time points where they are abnormal [8]. Again, this concept can be generalized by assuming each method employs a function that maps each edge in the graph to a real number, low values indicating unusual behaviour. A high-level definition of anomalous edge detection can be defined as follows:

   Definition 2: Anomalous edges (from [8]) Given G, the total edge set $E = \bigcup_{t=1}^{T} E_t$, and a specified scoring function $f : E \to \mathbb{R}$, the set of anomalous edges $E' \subseteq E$ is an edge set such that $\forall\, e' \in E',\ |f(e') - \hat{f}| > c_0$, where $\hat{f}$ is a summary statistic of the scores $f(e), \forall\, e \in E$.

3. **Anomalous Subgraphs** Finding subgraphs with irregular behaviour requires an approach different from the ones for anomalous vertices or edges, as enumerating every possible subgraph in even a single graph is an intractable problem [8]. We define the anomalous subgraphs as follows:

   Definition 3: Anomalous subgraphs (from [8]) Given G, a subgraph set $\mathrm{H} = \bigcup_{t=1}^{T} H_t$ where $H_t \subseteq G_t$ and a specified scoring function $f : H \rightarrow \mathbb{R}$, the set of anomalous subgraphs $H \subseteq H$ is a subgraph set such that $\forall h' \in H', \ |f(h') - \hat{f}| > c_0$, where $\hat{f}$ is a summary statistic of the scores $f(h), \ \forall h \in H$.

4. **Event and Change Detection** Event detection has attracted much interest in the data mining community, and it has a much broader scope compared to the previous three types of anomalies, aiming at identifying time points that are significantly different from the rest [8]. Isolated points in time where the graph is unlike the graphs at the previous and following time points represent events.

   Definition 4: Event detection (from [8]) Given G and a scoring function $f : G_t \rightarrow \mathbb{R}$, an event is defined as a time point t, such that $|f(G_t) - f(G_{t-1})| > c_0$ and $|f(G_t) - f(G_{t+1})| > c_0$. Now, we move on to the problem of change detection, which is complementary to event detection. It is important to note the distinction between event and change detection. While events represent isolated incidents, change points mark a point in time where the entire behaviour of the graph changes and the difference is maintained until the next change point [8].

   Definition 5: Change detection (from [8]) Given G and a scoring function $f : G_t \rightarrow \mathbb{R}$, a change is defined as a time point t, such that $|f(G_t) - f(G_{t-1})| > c_0$ and $|f(G_t) - f(G_{t+1})| \leq c_0$.

5.2 *Anomaly Detection on Dynamic Graphs Using Graph Analysis*

In this section we present an overview of graph analysis methods that are used for anomaly detection on dynamic graphs. These methods are divided according to three types of approaches: Vertex detection, subgraph detection, and change detection. Figure 2 shows how the methods are categorized. In addition, we present the limitations of these methods.

5.2.1 Methods for Vertex Detection

In [33] they proposed Incremental Local Evolutionary Outlier Detection (IcLEOD) method which illustrate that" group of vertices that belong to the same community are expected to exhibit similar behaviour. Probably this means that if at consecutive time steps one vertex in the community has a huge number of new edges added, the other vertices in the community would also have a huge number of new edges" [8]. If the remaining vertices in the community did not have new edges inserted, the vertex that

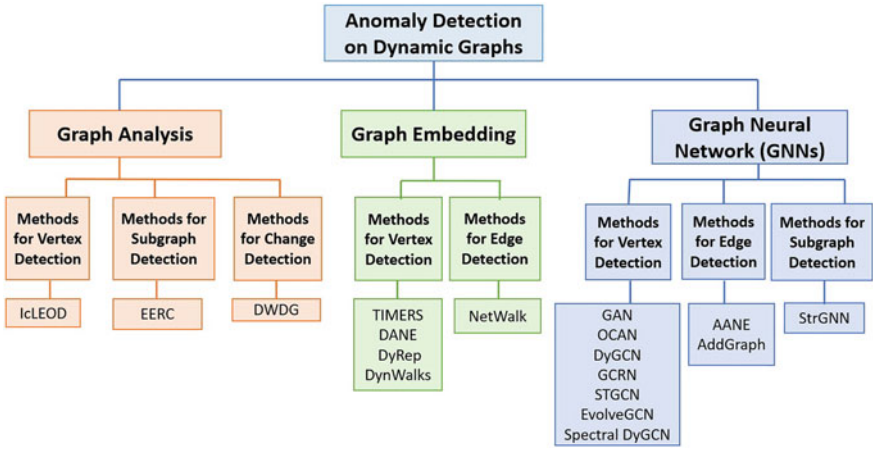


Fig. 2 Anomaly detection methods applied on dynamic graphs

did is detected as anomalous [34–37]. More regularly, "a vertex's corenet consists of itself and all the vertices within two hops that have a weighted path above a threshold value" [8]. "If the edge weight between two vertices is considered the strength of their connection, then intuitively the vertices connected with higher weight edges should be considered as a part of the same community" [8]. "Consequently, if a vertex has two neighbors removed, one connected with a high edge weight and the other connected with a low edge weight, then the removal of the vertex connected by the higher edge weight should have more impact" [8]. "At each time step, every vertex is first given an outlier score based on the change in its corenet, and the top-k outlier scores are then declared anomalous" [8].

5.2.2 Methods for Subgraph Detection

"Conversely, instead of finding changes, communities that are conserved, or stable, can be identified. Constructing multiple graphs at each time step based on different information sources, communities can be conserved across time and graphs" [8]. Graphs that act closely can be combined using clustering or prior knowledge. In [38] "the extreme events-related communities (EERC) method proposes that if a community is conserved across time steps and the graphs within its group, but has no corresponding community in any other group of graphs, then the community is defined as anomalous; two communities are considered corresponding communities if they have a certain percentage of their vertices in common". Unlike [38–40] that consider only the structure of the graph, in social network there is often more knowledge available. A cluster that has an evolution event is considered as an anomalous subgraph once the evolution event occurs [8].

5.2.3 Methods for Change Detection

Changes are detected by partitioning the streaming graphs into coherent segments based on the similarity of their partitioning (communities) [8]. The start of each segment shows a detected change. The segments are found online by comparing the vertex partitioning of the newest graph to the partitioning found for the graphs in the current growing segment [8]. Vertex partitioning can be achieved with many methods, but in [36] they propose "the community mining including community discovery and change-point detection on dynamic weighted directed graphs (DWDG) that is achieved using the relevance matrix computed by random walks with restarts and modularity maximization". When the partitioning of the new graph is much different from the current segment's, a new segment begins, and the time point for the new graph is marked as a detected change [8]. The similarity of two partitions is computed as their Jaccard coefficient, and the similarity of two partitioning is the normalized sum of their partition similarities [8].

5.2.4 Limitations

The main problem here is that vertices and edges are changing along the time dimension, and we need to capture the dependency between different graphs along the time dimension [5].

5.3 Anomaly Detection on Dynamic Graphs Using Graph Embedding

In this section we present an overview of graph embedding methods that are used for anomaly detection on dynamic graphs. These methods are divided according to two types of anomalies vertex detection, and edge detection. In addition, we present the limitations of these methods.

5.3.1 Methods for Vertex Detection

Following the analysis of encoding graph into an embedding space, after which anomaly detection is applied, dynamic graph representation methods have been studied in the more recent works. TIMERS [41] proposes an incremental Singular Value Decomposition (SVD) model for dynamic embedding, which only needs SVD at beginning, and incrementally updates them according to graph change. DNE [42] proposes a dynamic version of LINE, with only a few gradient descent processes to update the representation of current graph. DANE [43] proposes a graph embedding method in dynamic environment which updates the node embedding based on the

change in the adjacency matrix as well as in the attribute matrix via matrix perturbation. DyRep [44] ingests dynamic graph information in the form of association and communication events over time and updates the node representations as they appear in these events. DynWalks [45] incorporates the temporal information with traditional DeepWalk to capture the evolving properties in dynamic graphs. It updates nodes embeddings by sampling new walks which is highly related to the changes of graph [46]. Specifically, [47] introduces a flexible deep representation technique, NetWalk, to detect anomalous nodes in dynamic graphs using only the structure information [47]. It adopts an auto-encoder to learn node representations on the initial graph and incrementally updates them when new edges are added, or existing edges are deleted. For anomaly detection, NetWalk first adopts the streaming k-means clustering algorithm [48] to group existing nodes in the current time stamp into different clusters. Then, the anomaly score of each node is measured as its closest distance to the k clusters [48]. When node representations are updated, the cluster centers and anomaly scores are re-calculated accordingly.

5.3.2 Methods for Edge Detection

The intuition of graph representation-based techniques is to encode the dynamic graph structure information into edge representations and apply the traditional anomaly detection techniques to spot irregular edges. This is quite straightforward, but there remain vital challenges in generating/updating informative edge representations when the graph structure evolves. To mitigate this challenge, NetWalk [47] is also capable of detecting anomalous edges in dynamic graphs. Following the line of distance-based anomaly detection, NetWalk encodes edges into a shared latent space using node embeddings and then detect anomalies based on their distances to the nearest edge-cluster centers in the latent space. When new edges arrive or existing edges disappear, the node and edge representations will be updated based on random walks in the temporary graphs at each time stamp, after which the edge-cluster centers and edge anomaly scores are re-calculated [47]. Finally, the top-k farthest edges to edge-clusters are reported as anomalies.

5.3.3 Limitations

Although NetWalk can detect anomalies in dynamic graphs, it simply updates edge representations without considering the long/short-term node and graph structure evolving patterns. NetWalk [47] approach the anomalous node detection problem promisingly, but they respectively only study the structure or attributes. Considering the graph embedding problem in a dynamic manner is more corresponding to the real-world applications. More information could be collected when we study the dynamic features of graph. Although some research on dynamic graph embedding have been proposed they mainly focus on mining the pattern of graph involvement, they still ignore the efficiency issue. Most of these works perform traditional static

graph embedding in each snapshot of a dynamic graph, and then consider adding constraint or interactions between different snapshots [46]. Their methods consequently get better performance than previous static methods. However, generating new representation at each time is costly, since most traditional node representation methods are supposed to learn the embedding parameters by optimization process (e.g., gradient descent or matrix factorization). Repeating this progress at each time step brings about high complexity [46].

5.4 Anomaly Detection on Dynamic Graphs Using Graph Convolutional Neural Network

In this section we present an overview of graph convolutional neural graph methods that are used for anomaly detection on dynamic graphs. Graph neural graphs have been applied to perform reasoning on the dynamics of physical systems. Graph convolutional neural networks, which extend the convolutional neural networks to graph structure data, have been shown to improve the performance of graph classification and vertex level semi-supervised classification. A general framework for graph neural networks is proposed in [8]. The dynamic information has been proven to boost a variety of graph analytical tasks such as community detection, link prediction and graph embedding. Therefore, it has strong potential to advance graph neural networks by considering the dynamic nature of graphs, which calls for dedicated efforts [8]. These methods are divided according to two types of anomalies: vertex detection and edge detection. In addition, we present the limitations of these methods.

5.4.1 Methods for Vertex Detection

Generative adversarial networks (GAN) [49] have received extensive attention because of their impressive performance in capturing real data distribution and generating simulated data. In [50], they circumvented the fraudster detection problem using only the observed benign users' attributes. The basic idea is to seize the normal activity patterns and detect anomalies which behave in a significant different manner. The proposed method, OCAN [50], starts with the extraction of benign users' content features using their historical social behaviors (e.g., historical posts, posts' URL), for which this method is classified into the dynamic category. Dynamic Graph Convolutional Network (DyGCN) [46], which is an extension of ConvGNNs-based methods. They naturally generalize the embedding propagation scheme of ConvGNNs to dynamic setting in an efficient manner, which is to propagate the change along the graph to update node embeddings. The most affected nodes are first updated, and then their changes are propagated to the next nodes and lead to their update. Extensive experiments conducted on various dynamic graphs demonstrate that their model can update the node embeddings in a timesaving and performance-preserving way [46].

The dynamic graph prediction task is different from dynamic graph embedding, but there are still some similarities. Graph convolutional recurrent network (GCRN) [51] uses a Graph Convolutional Networks (ConvGNNs) to learn node embedding and feed it into the LSTM to learn dynamism. WD-ConvGNNs/CD-ConvGNNs [52] modifies the LSTM by incorporating it with ConvGNNs. Spatio-temporal Graph Convolutional Network (STGCN) [53] proposes a so-called ST-Conv blocks, which requires that the node features must be evolving over time. EvolveGCN [54] utilizes recurrent neural network (RNN) to evolve the ConvGNNs parameters, instead of the node embedding [46]. Spectral DyGCN [46] further upgrade the high-order mechanism above with acceptable time consumption. The high-order update mechanism ignores the global propagation of changing nodes. The changing information of 1-order nodes could not be propagated to all the nodes, which inevitably brings about the loss of accuracy.

5.4.2 Methods for Edge Detection

ConvGNNs can capture the structural information in edge detection. In AANE method [55] the authors demonstrate that the existence of anomalous edges in the training data prevents traditional ConvGNNs based models from capturing the real edge distribution, leading to sub-optimal detection performance. AANE solved the problem of detecting performance, by alleviating the negative impact of anomalous edges using the learned embeddings. AANE method iteratively updates the embeddings and detection results during training. In each training iteration, AANE generates node embeddings Z through ConvGNNs layers and learns an indicator matrix I to spot potential anomalous edges. AANE spots the top-k edges with lowest probabilities as anomalies [55]. Thus, an edge uv is identified as anomalous when its predicted probability is less than the average of all links associated with the node u by a predefined threshold. An alternative semi-supervised model, AddGraph, comprises a ConvGNNs and Gated Recurrent Units (GRU) with attention to capture more representative structural information from the temporal graph in each time stamp and the dependencies between them, respectively [13]. The hidden state of the nodes at each timestamp are used to calculate the anomalous probabilities of an existing edge and a negative sampled edge, and then feed them to a margin loss.

5.4.3 Methods for Subgraph Detection

In this section, we introduce a method for subgraph detection. Previous graph embedding based techniques have been frequently interesting on learning good node representations, whereas highly ignoring the sub-graph structural modifications related to the spot nodes in dynamic graphs. StrGNN [56] is an end-to-end structural temporal Graph Neural Network model for detecting anomalous edges in dynamic graphs. In precise, they first extract the hop enclosing subgraph centered on the spot edge and introduce the node labeling function to describe the role of each node in the [56].

Afterward, they drag graph convolution operation and sort pooling layer to extract the fixed-size feature from every snapshot/timestamp.

6 Open Challenges

Detecting anomalies using graph convolutional neural network is an important research direction, which leverages multi-source, multi-view features extracted from both content and structure for anomaly sample analysis and detection. It plays an important role in cyber security, but it still needs a lot of effort in the field due to the multiple issues from data, model and task. The future works are mainly lying in three perspectives: dynamic graphs, anomaly detection and graph machine learning.

Firstly, from dynamic graph learning perspective, there are two challenges [1]: Challenge 1 is the lack of raw attribute information on most dynamic graphs. Due to the explosive demand for data volume of time evolving attributes or the inaccessible attribute evolved by privacy problem, it is difficult to construct attribute information to describe each node from the mainstream raw dynamic graph datasets. This leads to the need for an effective encoding method to represent evolving nodes. Challenge 2 is the difficulty of learning discriminative information from dynamic graphs where structural knowledge and temporal knowledge are coupled [1]. The idea is that both structural and temporal factors should be considered simultaneously when making the agreement, which raises a challenge in learning such coupled information.

Secondly, from an anomaly analysis perspective, there are still a lot of research questions [2]. How to define and identify the anomalies in the graph in the different tasks? How to effectively convert the large-scale raw data to the graph? How to effectively leverage the attributes? How to model the dynamic during the graph construction? How to keep the heterogeneity during the graph construction? Recently, due to data-specific and task-specific issues, the applications of GNN-based anomaly detection are still limited. There is still a lot of potential scenarios that can be applied.

Finally, from a graph machine learning perspective, lots of issues need to be addressed [2]. How to model the graph? How to represent the graph? How to leverage the context? How to fuse the content and structure features? Which part of the structure to capture, local or global? How to provide the model explainability? How to protect the model from adversarial attacks? How to overcome the time–space scalability bottleneck. Recently, lots of contributions have been made from the machine learning perspective. However, due to the unique characteristics of the anomaly detection problem, which GNNs to use and how to apply GNNs are still critical questions. Further work will also benefit from the new findings and new models in the graph machine learning community.

7 Conclusion

Due to the complex relationships between real-world objects and recent approaches for analyzing graph (especially graph convolutional neural networks), graph anomaly detection with deep learning is currently at the forefront of anomaly detection. In this chapter we review the methods for graph anomaly detection with modern deep learning approaches. We divided these approaches between graph analysis, graph embedding and graph convolutional neural network. These approaches can be applied on static and dynamic graphs. For detecting anomalies on static graphs, we present graph analysis, graph embedding and graph convolutional neural network on plain static graphs and attribute static graphs. For detecting anomalies on dynamic graphs, we present graph analysis, graph embedding and graph convolutional neural network according to the types of graph anomalies they can detect as: (1) anomalous vertex detection; (2) anomalous edge detection; (3) anomalous subgraph detection and finally, (4) anomalous event and change detection. Clear summarizations and comparisons between different works are given, providing a complete and thorough picture of current work and the progress of graph anomaly detection as a field. Moreover, to push forward future research in this area, we provide open challenges for anomaly detection on dynamic graphs using graph neural networks.

References

1. Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q.Z., Xiong, H., Akoglu, L.: A comprehensive survey on graph anomaly detection with deep learning. IEEE Trans. Knowl. Data Eng. (2021)
2. Shen Wang, P.S.Y.: Chapter 26 Graph neural networks in anomaly detection, pp. 558–578. https://graph-neural-networks.github.io/static/file/chapter26.pdf
3. Xu, M.: Understanding graph embedding methods and their applications. arXiv:2012.08019 (2020)
4. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2020)
5. Zhou, R., Zhang, Q., Zhang, P., Niu, L., Lin, X.: Anomaly detection in dynamic attributed networks. Neural Comput. Appl. **33**(6), 2125–2136 (2021)
6. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: extending highdimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
7. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
8. Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C., Samatova, N.F.: Anomaly detection in dynamic networks: a survey. Wiley Interdiscip. Rev. Comput. Stat. **7**(3), 223–247 (2015)
9. Li, Q., Han, Z., Wu, X.-M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
10. Gupta, A., Matta, P., Pant, B.: Graph neural network: current state of art, challenges and applications. Mater. Today Proc. (2021)

11. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data Min. Knowl. Disc. **29**(3), 626–688 (2015)
12. Wang, H., Wu, J., Hu, W., Wu, X.: Detecting and assessing anomalous evolutionary behaviors of nodes in evolving social networks. ACM Trans. Knowl. Discov. Data (TKDD) **13**(1), 1–24 (2019)
13. Zheng, L., Li, Z., Li, J., Li, Z., Gao, J.: Addgraph: anomaly detection in dynamic graph using attention-based temporal GCN. IJCAI, 4419–4425 (2019)
14. Chen, H., Yin, H., Sun, X., Chen, T., Gabrys, B., Musial, K.: Multi-level graph convolutional networks for cross-platform anchor link prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1503–1511 (2020)
15. Silva, J., Willett, R.: Hypergraph-based anomaly detection of highdimensional co-occurrences. IEEE Trans. Pattern Anal. Mach. Intell. **31**(3), 563–569 (2008)
16. Li, G., Jung, J.J.: Entropy-based dynamic graph embedding for anomaly detection on multiple climate time series. Sci. Rep. **11**(1), 1–10 (2021)
17. Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: spotting anomalies in weighted graphs. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 410–421. Springer (2010)
18. Ding, Q., Katenka, N., Barford, P., Kolaczyk, E., Crovella, M.: Intrusion as (anti) social communication: characterization and detection. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 886–894 (2012)
19. Hooi, B., Song, H.A., Beutel, A., Shah, N., Shin, K., Faloutsos, C.: Fraudar: Bounding graph fraud in the face of camouflage. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 895–904 (2016)
20. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 1025–1035 (2017)
21. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584 (2017)
22. Liu, N., Huang, X., Hu, X.: Accelerated local anomaly detection via resolving attributed networks. IJCAI, 2337–2343 (2017)
23. Wu, L., Hu, X., Morstatter, F., Liu, H.: Adaptive spammer detection with sparse group modeling. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 11 (2017)
24. Wang, Z., Lan, C.: Towards a hierarchical Bayesian model of multi-view anomaly detection. IJCAI, 2420–2426 (2020)
25. Miao, K., Shi, X., Zhang, W.-A.: Attack signal estimation for intrusion detection in industrial control system. Comput. Secur. **96**, 101926 (2020)
26. Jie, F., Wang, C., Chen, F., Li, L., Wu, X.: Block-structured optimization for anomalous pattern detection in interdependent networks. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 1138–1143. IEEE (2019)
27. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv:1609.02907 (2016)
28. Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: Proceedings of the 2019 SIAM International Conference on Data Mining, pp. 594–602. SIAM (2019)
29. Bandyopadhyay, S., Vivek, S.V., Murty, M.: Outlier resistant unsupervised deep architectures for attributed network embedding. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 25–33 (2020)
30. Li, Y., Huang, X., Li, J., Du, M., Zou, N.: Specae: spectral autoencoder for anomaly detection in attributed networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 2233–2236 (2019)
31. Wang, J., Wen, R., Wu, C., Huang, Y., Xion, J.: Fdgars: Fraudster detection via graph convolutional networks in online app review system. In: Companion Proceedings of the 2019 World Wide Web Conference, pp. 310–316 (2019)

32. Zhang, S., Yin, H., Chen, T., Hung, Q.V.N., Huang, Z., Cui, L.: GCN-based user representation learning for unifying robust recommendation and fraudster detection. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 689–698 (2020)
33. Ji, T., Yang, D., Gao, J.: Incremental local evolutionary outlier detection for dynamic social networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 1–15. Springer (2013)
34. Saligrama, V., Chen, Z.: Video anomaly detection based on local statistical aggregates. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2112–2119. IEEE (2012)
35. Tran, L., Navasca, C., Luo, J.: Video detection anomaly via low-rank and sparse decompositions. In: 2012 Western New York Image Processing Workshop, pp. 17–20. IEEE (2012)
36. Duan, D., Li, Y., Jin, Y., Lu, Z.: Community mining on dynamic weighted directed graphs. In: Proceedings of the 1st ACM International Workshop on Complex Networks Meet Information & Knowledge Management, pp. 11–18 (2009)
37. Tantipathananandh, C., Berger-Wolf, T.Y.: Finding communities in dynamic social networks. In: 2011 IEEE 11th International Conference on Data Mining, pp. 1236–1241. IEEE (2011)
38. Chen, Z., Hendrix, W., Guan, H., Tetteh, I.K., Choudhary, A., Semazzi, F., Samatova, N.F.: Discovery of extreme events-related communities in contrasting groups of physical system networks. Data Min. Knowl. Disc. **27**(2), 225–258 (2013)
39. Chen, Z., Hendrix, W., Samatova, N.F.: Community-based anomaly detection in evolutionary networks. J. Intell. Inf. Syst. **39**(1), 59–85 (2012)
40. Araujo, M., Papadimitriou, S., Gu¨nnemann, S., Faloutsos, C., Basu, P., Swami, A., Papalexakis, E.E., Koutra, D.: Com2: fast automatic discovery of temporal ('comet') communities. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 271–283. Springer (2014)
41. Zhang, Z., Cui, P., Pei, J., Wang, X., Zhu, W.: Timers: error-bounded SVD restart on dynamic networks. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
42. Du, L., Wang, Y., Song, G., Lu, Z., Wang, J.: Dynamic network embedding: an extended approach for skip-gram based network embedding. IJCAI **2018**, 2086–2092 (2018)
43. Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., Liu, H.: Attributed network embedding for learning in a dynamic environment. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 387–396 (2017)
44. Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: Dyrep: learning representations over dynamic graphs. In: International Conference on Learning Representations (2019)
45. Hou, C., Zhang, H., Tang, K., He, S.: DynWalks: global topology and recent changes awareness dynamic network embedding. arXiv:1907.11968 (2019)
46. Cui, Z., Li, Z., Wu, S., Zhang, X., Liu, Q., Wang, L., Ai, M.: DYGCN: Dynamic graph embedding with graph convolutional network arXiv:2104.02962 (2021)
47. Yu, W., Cheng, W., Aggarwal, C.C., Zhang, K., Chen, H., Wang, W.: Netwalk: a flexible deep embedding approach for anomaly detection in dynamic networks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2672–2681 (2018)
48. Zhang, Y., Tangwongsan, K., Tirthapura, S.: Streaming k-means clustering with fast queries. In: 2017 IEEE 33rd International Conference on Data Engineering (ICDE), pp. 449–460. IEEE (2017)
49. Liu, Z., Dou, Y., Yu, P.S., Deng, Y., Peng, H.: Alleviating the inconsistency problem of applying graph neural network to fraud detection. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1569–1572 (2020)
50. Zheng, P., Yuan, S., Wu, X., Li, J., Lu, A.: One-class adversarial nets for fraud detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1286–1293 (2019)
51. Seo, Y., Defferrard, M., Vandergheynst, P., Bresson, X.: Structured sequence modeling with graph convolutional recurrent networks. In: International Conference on Neural Information Processing, pp. 362–373. Springer (2018)

52. Manessi, F., Rozza, A., Manzo, M.: Dynamic graph convolutional networks. Pattern Recognit. **97**, 107000 (2020)
53. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv: 1810.00826 (2018)
54. Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T., Leiserson, C.: Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5363–5370 (2020)
55. Duan, D., Tong, L., Li, Y., Lu, J., Shi, L., Zhang, C.: AANE: Anomaly aware network embedding for anomalous link detection. In: 2020 IEEE International Conference on Data Mining (ICDM), pp. 1002–1007. IEEE (2020)
56. Cai, L., Chen, Z., Luo, C., Gui, J., Ni, J., Li, D., Chen, H.: Structural temporal graph neural networks for anomaly detection in dynamic graphs. arXiv:2005.07427 (2020)

Secure Medical Data Sharing Through Blockchain and Decentralized Models

Francesco Colace, Massimo De Santo, Francesco Marongiu, Domenico Santaniello, and Alfredo Troiano

Abstract The global health system has been strongly impacted by the events caused by the Covid-19 pandemic, highlighting and amplifying several critical issues. To contrast a global pandemic, it is necessary to have tools that can offer an integral vision of the situation to act in a timely manner on any possible case. In this regard, technologies based on decentralized systems, such as Blockchain, can be helpful, as they are global, real-time, and can be implemented to respect people's privacy. This work aims to investigate the use of decentralized networks to develop a system that can solve the problem of management and collaborative analysis of sensitive data. A general framework has been proposed that is mainly dedicated to healthcare organizations, such as hospitals and research centres, which first had to face the difficult situation caused by the Covid-19 pandemic. It has been highlighted how decentralized systems can effectively improve communications between these entities, promoting the fair exchange of knowledge in real-time without compromising citizens' privacy.

Keywords Big data · Blockchain · Digital storage · E-Health · Smart contract · Trusted data sharing · Covid-19

F. Colace · M. De Santo · F. Marongiu · D. Santaniello (✉)
DIIn, University of Salerno, Fisciano, SA, Italy
e-mail: dsantaniello@unisa.it

F. Colace
e-mail: fcolace@unisa.it

M. De Santo
e-mail: desanto@unisa.it

F. Marongiu
e-mail: fmarongiu@unisa.it

A. Troiano
NetCom Group, Napoli, Italy
e-mail: a.troiano@netcomgroup.eu

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

N. Nedjah et al. (eds.), *Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities*, Studies in Computational Intelligence 1030,
https://doi.org/10.1007/978-3-030-96737-6_13

1 Introduction

One of the most significant sources of knowledge is certainly represented by big data: large amounts of information that, if analyzed properly, can reveal valuable information [10, 14]. This information is particularly evident in a global pandemic situation, where the number of infections, geographic locations, patients' health, etc. are crucial to fighting the spread of the epidemic [26, 32].

Knowing the virus's movements in real-time it is possible, for example, to identify critical areas of high contagion to prevent the spread of the virus [4]. However, collecting and analyzing this type of information brings several problems. First, this information must be stored globally and shared among all specialized agencies, while detailed access to all information must be selective to protect people's privacy [18].

Sharing this information is another essential aspect. Especially when it comes to health data, it is crucial to ensure its security in all aspects. Also, in exceptional cases such as a global pandemic, the ability to access this data in real-time from multiple organizations both regionally and internationally becomes essential [9, 30].

Thanks to a decentralized approach, it is possible to design scalable systems with a high degree of reliability, capable of guaranteeing all the requirements described above [17]. Therefore, data are distributed within a global network that can maintain unaltered properties and ensure information security. Peer-2-Peer networks can be used to achieve this result. In this type of architecture, information is exchanged autonomously by network participants and through a distributed consensus protocol validated in real-time. Through the support of cryptographic models, it is possible to achieve certification and immutability of data and appropriate management of sensitive information [27].

Blockchain and decentralized file system technologies (i.e., InterPlanetary FileSystem—IPFS) [27] are well suited to develop architectures that meet the requirements in the analysis [1, 12, 15, 21]. The aim of the paper is, therefore, to create a framework able to guarantee:

Security against any disasters: Since the data are distributed and replicated within a global network, there is no risk of having a single point of failure that could compromise them irreversibly.

Privacy: Through the support of modern cryptographic techniques, it is possible to guarantee data ownership to specific users and hide sensitive information from outsiders [2, 13, 22]. This approach is extremely useful in the healthcare sector.

Through *Real-time information sharing*, inherent in decentralized systems, organizations of various types can fair collaborate [20].

The Covid-19 pandemic indeed highlighted the need to develop systems other than traditional ones. Since the beginning, Hospitals and Research Centers have been faced with managing a large amount of sensitive and unstructured information, difficult to analyze and are worldwide with other institutions to deal with the spread of the virus [24, 25].

2 Related Works

Several approaches have been proposed in the literature for health data management using decentralized systems.

Azaria et al. [8] propose MedRec, a decentralized management system able to handle electronic medical records using blockchain technology. MedRec manages authentication, confidentiality, accountability, and data sharing. The system allows users to control their medical information through a blockchain network consisting of several medical stakeholders securing the web utilizing a Proof-of-Work consensus protocol.

Christodoulou et al. [11] also focused their research on people, giving them exclusive data ownership. They proposed a decentralized system for direct information exchange between doctors and patients using cryptographic techniques and Smart Contracts.

However, posing the User at the center of this information exchange does not apply to a real-world use case. Generally, information of this type has an entirely different meaning to users than organizations, such as Hospitals or Research Centers. Users are only interested in their health status, whereas a Hospital Entity uses data from all patients to derive additional knowledge for the benefit of the community. A prime example is the Covid-19 Pandemic. To prevent contagions and disease spread, certified and competent Institutions must have complete control over the data to provide the best possible instructions; this is not possible in [8, 11].

Hasan et al. [16] proposed an interesting approach in which IPFS and Blockchain events are used. In this model, data is selectively encrypted and shared using an asymmetric key encryption mechanism. However, the use of the Blockchain is marginal as only events are utilized while storing any data is given over to IPFS. The letter does not include itself to spread data across the network to decentralize and secure, this can only be done using FileCoin,[1] thus additional costs. Finally, their proposed sharing system requires a module dedicated to exchanging keys and references to files; the presence of this module partially reduces the effectiveness of Blockchain advantages.

The proposed methodology takes up what is currently the State of the Art in this field, improving its structure so that it can be used in real-life scenarios. The framework is essentially divided into three parts:

The Blockchain is used to store metadata of information in a secure, immutable, and sequential way.

Decentralized storage is used to store encrypted information without impacting the Blockchain performance with a high degree of scalability.

In addition, the Blockchain takes care of the access and sharing of metadata among the different system users without the intervention of other external modules or authorities.

[1] https://docs.filecoin.io/about-filecoin/ipfs-and-filecoin/#data-storage-incentives.

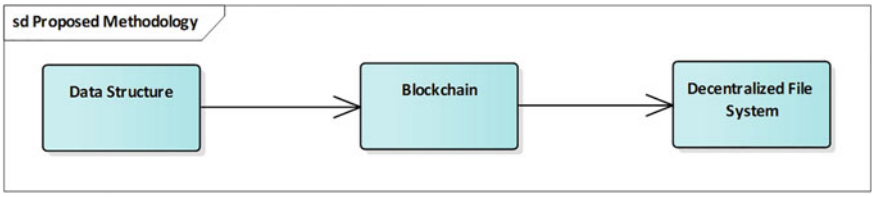


Fig. 1 Modules of the proposed methodology

3 Proposed Methodology

The framework is proposed as a solution to the more general problem of storing a large amount of information in a secure, scalable way that allows rapid data sharing while certifying provenance, immutability, confidentiality, and facilitating collaboration in the study and analysis of data. For this reason, the system is based on three main concepts: Blockchain; Decentralized Storage; Symmetric and Asymmetric Key Encryption [29] (Fig. 1).

The first step in creating the architecture is to organize the information to be encoded in a defined structure. Then, using Blockchain and Decentralized Storage Systems, meta-data and data are saved with appropriate cryptographic techniques, respectively.

To realize an experimental prototype, the model has been implemented using the Ethereum Blockchain and EVM (Ethereum Virtual Machine) [3] for the Smart Contract. The decentralized file system module was instead implemented using IPFS.

The different modules of which the architecture is composed will be specified in detail below.

By convention, the leading actor in the system will be referred to by the generic term *User*. In this particular use case, a User will be a single organization that has to store and organize a massive amount of data collected by its infrastructure.

4 Data Structure

Medical data is almost always unstructured, meaning that it is often difficult to trace all the information back to a standard structure for computer storage [19]. However, for the model to work, it is necessary to create a data structure that can accommodate health data as entirely as possible according to a well-defined structure, which is at the same time dynamic (Fig. 2).

A graph structure has been proposed, where each node represents the smallest piece of information desired to be saved within the system.

Each piece of information is then identified with a unique address (**ContentURI**) generated by the user who wishes to upload a file into the system. Using this structure,

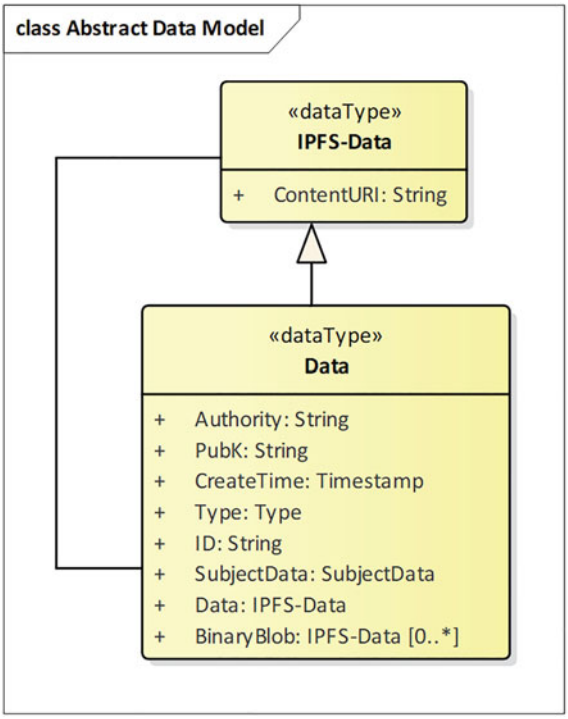


Fig. 2 Abstract data structure definition

it is possible to describe information without a predefined data schema dynamically. In addition, the *ContentURI* field allows data to be structured semantically as well.

5 Blockchain

All metadata files are contained within the Blockchain. In particular: the date of insertion; the owner (intended as the User who uploaded the file); whether the data can change over time; and the reference to the location of the binary data on the decentralized file system. Due to their structure, Blockchain systems are not scalable for storing large files [31], therefore, only the meta-data of the information was chosen to be saved in this module, Moreover, operations on Blockchain are expensive both in economic and computational terms, so it's common practice to keep these operations as light as possible.

This module was implemented by using the concept of Smart Contract [33], which allows storing information guaranteed by the Blockchain. This Smart Contract acts as a point of information exchange between the various Users. In this way, no other module is needed for communication, and the parties involved can interact independently (Fig. 3).

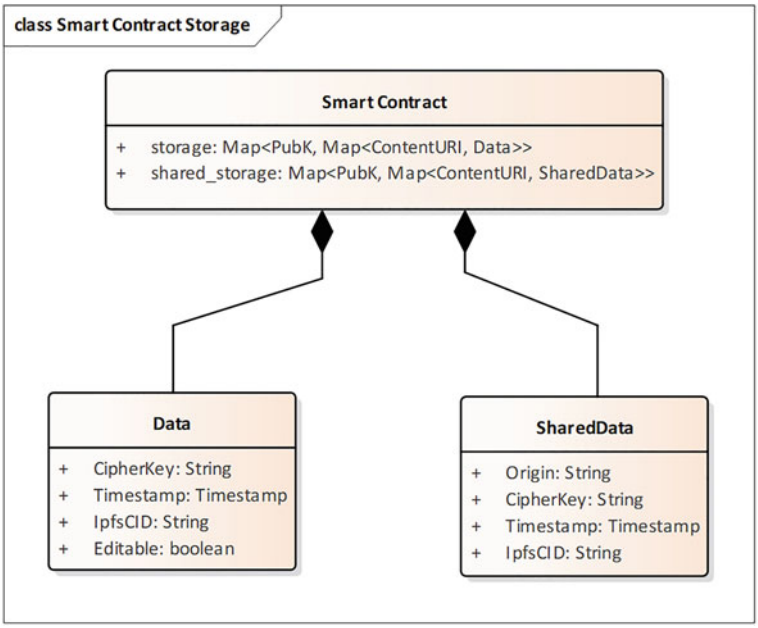


Fig. 3 Blockchain storage model

The memory of the Smart Contract has been divided into two distinct parts. Hash Tables structures have been chosen for data storage; each User (represented by a public key) has space where can uniquely associate a *ContentURI* to a particular meta-data. Like the previous one, the second part is a separate storage space dedicated exclusively to sharing meta-data between different users and exchanging information.

6 Data Storage and Management

Store Data. Once the data has been organized into the Graph structure previously described, the information is sent to the different modules in the following way, assuming that the User is already registered into the system and has its own Public (`<PubK>`) and Private Key (`<PrivK>`):

Before storing a data item, a User must first encrypt it with a uniformly randomly generated symmetric key `<SPrivK>`. The generation operation must be repeated for each entry. The binary data (encrypted according to a standard block cipher) is uploaded to IPFS, which returns the Content Identifier (**CID**) for addressing. At this point, the data is associated with a unique URI (*ContentURI*) generated by the User so that it is meaningful to describe the content of the data.

The symmetric key <SPrivK> is encapsulated with further encryption using its public key <PubK> and finally destroyed to prevent it from being recovered by an unauthorized party. The resulting cipher takes the name **CipherKey**.

The information is sent to Smart Contract, which takes care of storing it:

Address: User's public key.
ContentURI: URI that uniquely describes the data.
CipherKey: Symmetric key, used to encrypt the data, encapsulated.
CipherAlgorithm: Algorithm used for symmetric data encryption.
IpfsCID: CID returned by IPFS when the file is uploaded.
Editable: Boolean value indicating whether the CID field can be updated in the future or not.

In the medical field, some information is subject to change over time. For this reason, the ability to update some CID references associated with ContentURIs marked as editable has been provided. Some Decentralized File Systems, such as IPFS, do not allow deletion of files once they have been inserted; obsolete versions of the data will remain present in the network as history.

The framework does not provide any particular type of algorithm for the encryption mechanism. It is the user who can decide which technique to use to the kind of information. As with hash functions on IPFS,[2] adopting a design that can be easy to evolve was deemed appropriate (Fig. 4).

Retrieve data. Retrieving data from decentralized services is done similarly:

Users can request from the Smart Contract the tuple of information (CypherKey, CipherAlgorithm, IpfsCID) by giving as input the pair of values (PubK, ContentURI).

At this point, the symmetric key must be extracted from the CypherKey value, and decryption is performed using <PrivK> to trace back to the symmetric key <SPrivK>.

The encrypted data is downloaded from IPFS using its addressing CID, decrypted using <SPrivK>, which for security reasons is immediately destroyed (Fig. 5).

Share Data. Sharing can only be done after the data has been uploaded to Smart Contract. Therefore, it is assumed that the user who wants to share a piece of information is already in possession of <SPrivK>, IpfsCID, and ContentURI and the User's public key to whom he wants to view his data. Information exchange is made by transferring the <SPrivK> information between Users, and no other private information is exchanged during the sharing process, particularly the User's private keys.

The following information is sent to the Smart Contract:

SharedAddress: Public key of the User with whom the data is to be shared.
ContentURI: URI that uniquely describes the data.
CipherKey: <SPrivK>used to encrypt the data, encapsulated using *SharedAddress* as the encryption key.

[2] https://multiformats.io/multihash/.

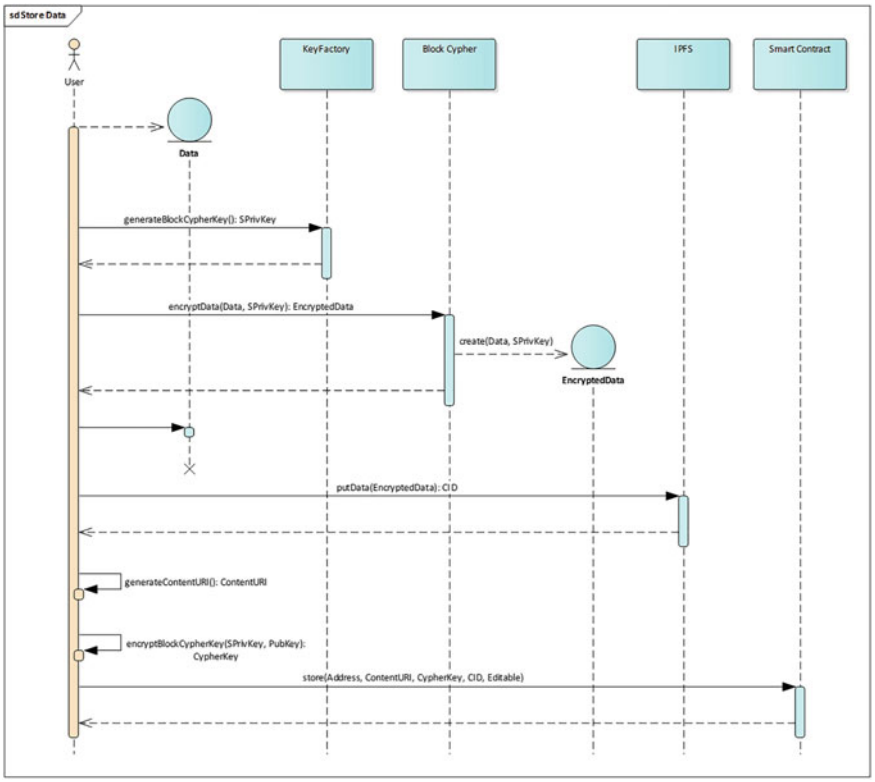


Fig. 4 Store data sequence diagram

CipherAlgorithm: Algorithm used for data encryption.
IpfsCID: CID returned by IPFS when uploading the file.

The User to whom the files are shared will use the steps described in the previous paragraph to access the data.

Some fields are automatically managed by Smart Contract to ensure data security and certification:

The *timestamp* fields are always populated with the time of the block in which the transaction is made. This field allows for certification of data entry over time.

When sharing data, the origin field is automatically set as the User's public key making the transaction. This step allows certifying the tracking of the data (Fig. 6).

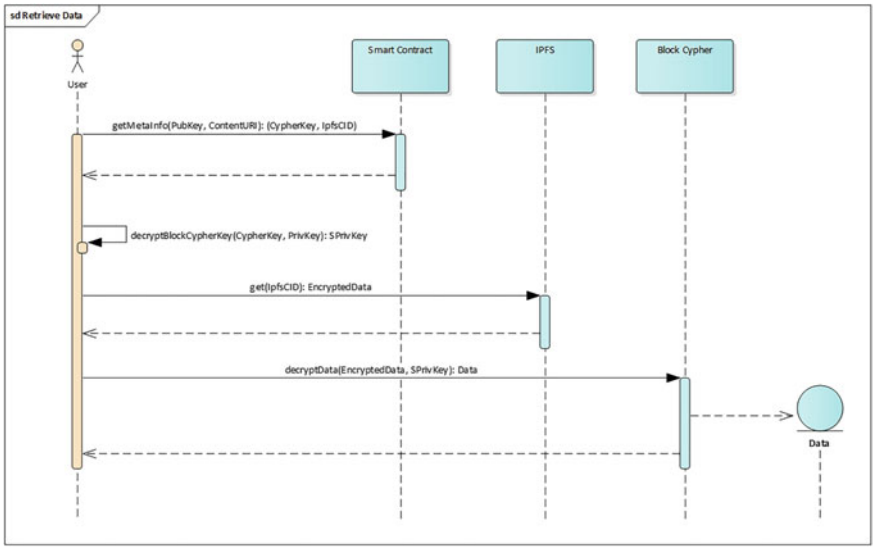


Fig. 5 Retrieve data sequence diagram

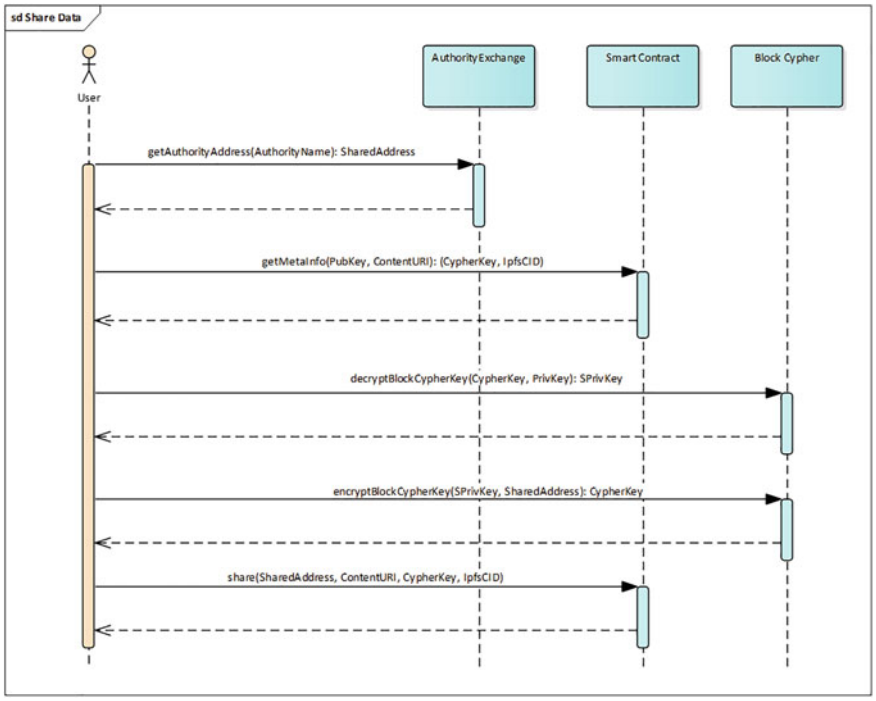


Fig. 6 Share data sequence diagram

7 Implementation into Ethereum Virtual Machine

The system has been implemented on Ethereum Blockchain via Smart Contract using Solidity language.

Main Data Structures in the Smart Contract

```
    struct Data {
        string cypherkey;
        string cypheralgorithm;
        uint256 timestamp;
        string ipfscid;
        bool editable;
        bool isValue;
    }
    struct SharedData {
        address origin;
        string cypherkey;
        string cypheralgorithm;
        uint256 timestamp;
        string ipfscid;
        bool isValue;
    }
    mapping(address => mapping(bytes32 => Data)) dataS-
torage;
    mapping(address => mapping(bytes32 => SharedData))
sharedDataStorage;
```

The main structures of the Smart Contract are represented by the hash tables that contain the data. A public key is uniquely associated with a second hash table to which a ContentURI can be associated with metadata.

Store Function

```
function storeData(bytes32 contentUri, string storage
cypherkey, string storage cypheralgorithm,string storage
ipfscid, bool editable) public returns(bool success) {
       //Deny storage if contentUri is already used

require(!dataStorage[msg.sender][contentUri].isValue);
       //Store values in data structure
       dataStorage[msg.sender][contentUri].cypherkey =
cypherkey;

dataStorage[msg.sender][contentUri].cypheralgorithm =
cypheralgorithm;
       dataStorage[msg.sender][contentUri].ipfscid =
ipfscid;
       dataStorage[msg.sender][contentUri].timestamp =
now;

       dataStorage[msg.sender][contentUri].editable =
editable;

       dataStorage[msg.sender][contentUri].isValue =
true;
       //Emit Blockchain event when new data is inserted
       emit DataInsert(msg.sender, contentUri);
       return true;
   }
```

To reduce costs and increase the efficiency of thes Smart Contract execution, all functions have been realized to perform only essential operations, trying to optimize statements and functions.

Blockchain Events list

```
    event DataInsert(address authority, bytes32 conten-
tUri);
   event DataModified(address authority, bytes32 conten-
tUri);
   event DataShared(bytes32 contentUri, address from,
address to);
```

Operations of insertion, edit, or share are associated with an event, respectively. An event is a signal propagated throughout the peer-2-peer network, other actors can access these events in real-time, so they are essential for further development of decentralized applications.

8 Conclusions

In conclusion, the work described in this paper aims to investigate in-depth one of the possible applications of Blockchain, and more generally of decentralized systems, for the storage and exchange of sensitive information at the institutional level. Indeed, we succeeded in realizing a scalable structure to collect data easily shared within a well-defined network. The system obtained is therefore independent of any central organization. This collaborative aspect of the framework allows it to be used in extraordinary situations, such as, for example, a global pandemic. The encryption techniques used are safe because they follow the current standards without changing the algorithms or compositions. Moreover, they have been used so as not to create dependencies in key usages. In the proposed framework, each cryptographic primitive uses its private key.

This work opens to multiple future developments. New technology could be used for the framework implementation. The most critical aspect of this system is the cost of maintaining the public network on which it relies. Moreover, operations on the Blockchain have a variable and often unpredictable price. For this reason, using public technologies may not be feasible in a real use-case scenario. It would therefore be interesting to study alternative implementations of currently experimental technologies, such as, for example, Blockchain with integrated decentralized storage.[3] Another aspect that could be investigated would be a link to a Public Key Infrastructure (PKI) to manage the public keys of all users, also using decentralized models and approaches [5–7]. Smart Contract events also represent a good starting point for future development, as they can be utilized to build decentralized applications able to operate on the data in real-time [28], such as building indexing services, notification services, or big data analysis services that require the certification of input data [23].

References

1. Abbas, K., Tawalbeh, L.A., Rafiq, A., Muthanna, A., Elgendy, I.A., Abd El-Latif, A.A.: Convergence of Blockchain and IoT for secure transportation systems in smart cities. Secur. Commun. Netw. **2021**, 1–13 (2021). https://doi.org/10.1155/2021/5597679
2. Abd El-Latif, A.A., Abd-El-Atty, B., Mehmood, I., Muhammad, K., Venegas-Andraca, S.E., Peng, J.: Quantum-inspired Blockchain-based cybersecurity: securing smart edge utilities in IoT-based smart cities. Inf. Process. Manage. **58**(4),(2021). https://doi.org/10.1016/j.ipm.2021.102549
3. Abou-Nassar, E.M., Iliyasu, A.M., El-Kafrawy, P.M., Song, O.-Y., Bashir, A.K., El-Latif, A.A.A.: DITrust Chain: towards Blockchain-based trust models for sustainable healthcare IoT systems. IEEE Access **8**, 111223–111238 (2020). https://doi.org/10.1109/ACCESS.2020.2999468
4. Agbehadji, I.E., Awuzie, B.O., Ngowi, A.B., Millham, R.C.: Review of big data analytics, artificial intelligence and nature-inspired computing models towards accurate detection of

[3] NeoFS: public distributed decentralized object storage: https://fs.neo.org/.

COVID-19 pandemic cases and contact tracing. Int. J. Environ. Res. Public Health **17**(15) (2020). https://doi.org/10.3390/ijerph17155330
5. Al-Bassam, M.: SCPKI. In: Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts (2017). https://doi.org/10.1145/3055518.3055530
6. Ali, M., Nelson, J., Shea, R., Freedman, M.J.: Blockstack: a global naming and storage system secured by blockchains. In: Proceedings of the 2016 USENIX Annual Technical Conference, USENIX ATC 2016 (2016)
7. Al-Qerem, A., Alauthman, M., Almomani, A., Gupta, B.B.: IoT transaction processing through cooperative concurrency control on fog–cloud computing environment. Soft Comput. **24**(8) (2020). https://doi.org/10.1007/s00500-019-04220-y
8. Azaria, A., Ekblaw, A., Vieira, T., Lippman, A.: MedRec: using blockchain for medical data access and permission management. In: 2016 2nd International Conference on Open and Big Data (OBD) (2016). https://doi.org/10.1109/OBD.2016.11
9. Cantelli, G., Cochrane, G., Brooksbank, C., McDonagh, E., Flicek, P., McEntyre, J., Birney, E., Apweiler, R.: The European Bioinformatics Institute: empowering cooperation in response to a global health crisis. Nucleic Acids Res. **49**(D1) (2021). https://doi.org/10.1093/nar/gkaa1077
10. Chen, H.-C., Chiang, R.H.L., Storey, V.C.: Business intelligence and analytics: from big data to big impact. MIS Q. **36**(4) (2012). https://doi.org/10.2307/41703503
11. Christodoulou, K., Christodoulou, P., Zinonos, Z., Carayannis, E.G., Chatzichristofis, S.A.: Health information exchange with blockchain amid Covid-19-like pandemics. In: 2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS) (2020). https://doi.org/10.1109/DCOSS49796.2020.00071
12. Dolgui, A., Ivanov, D., Potryasaev, S., Sokolov, B., Ivanova, M., Werner, F.: Blockchain-oriented dynamic modelling of smart contract design and execution in the supply chain. Int. J. Prod. Res. **58**(7) (2020). https://doi.org/10.1080/00207543.2019.1627439
13. Esposito, C., Ficco, M., Gupta, B.B.: Blockchain-based authentication and authorization for smart city applications. Inf. Process. Manage. **58**(2) (2021). https://doi.org/10.1016/j.ipm.2020.102468
14. Ghoneim, A., Muhammad, G., Amin, S.U., Gupta, B.: Medical image forgery detection for smart healthcare. IEEE Commun. Mag. **56**(4) (2018). https://doi.org/10.1109/MCOM.2018.1700817
15. Griggs, K.N., Ossipova, O., Kohlios, C.P., Baccarini, A.N., Howson, E.A., Hayajneh, T.: Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. J. Med. Syst. **42**(7) (2018). https://doi.org/10.1007/s10916-018-0982-x
16. Hasan, H.R., Salah, K., Jayaraman, R., Arshad, J., Yaqoob, I., Omar, M., Ellahham, S.: Blockchain-based solution for COVID-19 digital medical passports and immunity certificates. IEEE Access **8**,(2020). https://doi.org/10.1109/ACCESS.2020.3043350
17. Ichikawa, D., Kashiyama, M., Ueno, T.: Tamper-resistant mobile health using blockchain technology. JMIR MHealth UHealth **5**(7) (2017). https://doi.org/10.2196/mhealth.7938
18. Ienca, M., Vayena, E.: On the responsible use of digital data to tackle the COVID-19 pandemic. Nat. Med. **26**(4) (2020). https://doi.org/10.1038/s41591-020-0832-5
19. Lee, K.K.-Y., Tang, W.-C., Choi, K.-S.: Alternatives to relational database: comparison of NoSQL and XML approaches for clinical data storage. Comput. Methods Program. Biomed. **110**(1) (2013). https://doi.org/10.1016/j.cmpb.2012.10.018
20. Malek, M.S.B.A., Ahmadon, M.A.B., Yamaguchi, S., Gupta, B.B.: On privacy verification in the IoT service based on PN2. In: 2016 IEEE 5th Global Conference on Consumer Electronics, GCCE 2016 (2016). https://doi.org/10.1109/GCCE.2016.7800314
21. Mamta, D., Gupta, B.B., Li, K.C., Leung, V.C.M., Psannis, K.E., Yamaguchi, S.: Blockchain-assisted secure fine-grained searchable encryption for a cloud-based healthcare cyber-physical system. IEEE/CAA J. Autom. Sinica **8**(12) (2021). https://doi.org/10.1109/JAS.2021.1004003
22. Masud, M., Gaba, G.S., Alqahtani, S., Muhammad, G., Gupta, B.B., Kumar, P., Ghoneim, A.: A lightweight and robust secure key establishment protocol for internet of medical things in COVID-19 patients care. IEEE Internet Things J. (2020). https://doi.org/10.1109/JIOT.2020.3047662

23. Nguyen, G.N., le Viet, N.H., Elhoseny, M., Shankar, K., Gupta, B.B., El-Latif, A.A.A.: Secure blockchain enabled cyber–physical systems in healthcare using deep belief network with ResNet model. J. Parallel Distrib. Comput. **153** (2021). https://doi.org/10.1016/j.jpdc.2021.03.011
24. Nguyen, G.N., le Viet, N.H., Elhoseny, M., Shankar, K., Gupta, B.B., El-Latif, A.A.A.: Secure blockchain enabled Cyber–physical systems in healthcare using deep belief network with ResNet model. J. Parallel Distrib. Comput. **153**, 150–160 (2021). https://doi.org/10.1016/j.jpdc.2021.03.011
25. Park, S., Choi, G.J., Ko, H.: Information technology–based tracing strategy in response to COVID-19 in South Korea—privacy controversies. JAMA **323**(21) (2020). https://doi.org/10.1001/jama.2020.6602
26. Pham, Q.-V., Nguyen, D.C., Huynh-The, T., Hwang, W.-J., Pathirana, P.N.: Artificial intelligence (AI) and Big data for coronavirus (COVID-19) pandemic: a survey on the state-of-the-arts. IEEE Access **8** (2020). https://doi.org/10.1109/ACCESS.2020.3009328
27. Saroiu, S., Gummadi, K.P., Dunn, R.J., Gribble, S.D., Levy, H.M.: An analysis of internet content delivery systems. ACM SIGOPS Oper. Syst. Rev. **36**(SI) (2002). https://doi.org/10.1145/844128.844158
28. Sedik, A., Hammad, M., Abd El-Samie, F.E., Gupta, B.B., Abd El-Latif, A.A.: Efficient deep learning approach for augmented detection of Coronavirus disease. Neural Comput. Appl. (2021). https://doi.org/10.1007/s00521-020-05410-8
29. Sharifian, S., Safavi-Naini, R.: Information-theoretic Key Encapsulation and its Applications (2021)
30. Tewari, A., Gupta, B.B.: A lightweight mutual authentication protocol based on elliptic curve cryptography for IoT devices. Int. J. Adv. Intell. Paradigms **9**(2–3) (2017). https://doi.org/10.1504/IJAIP.2017.082962
31. Vukolić, M.: The quest for scalable blockchain fabric: proof-of-work vs. BFT replication (2016).https://doi.org/10.1007/978-3-319-39028-4_9
32. Wang, C.J., Ng, C.Y., Brook, R.H.: Response to COVID-19 in Taiwan. JAMA **323**(14) (2020). https://doi.org/10.1001/jama.2020.3151
33. Zou, W., Lo, D., Kochhar, P.S., Le, X.-B.D., Xia, X., Feng, Y., Chen, Z., Xu, B.: Smart contract development: challenges and opportunities. IEEE Trans. Softw. Eng. (2019). https://doi.org/10.1109/TSE.2019.2942301